AD-A097 375 MISSISSIPPI STATE UNIV MISSISSIPPI STATE ENGINEERING--ETC F/G 9/2
A STUDY OF AVIONICS TIME DIVISION MULTIPLEX BUS SIMULATION.(U)
DEC 80 M D CALHOUN, B KHATIRZAD AFOSR-80-0126
UNCLASSIFIED MSSU-EIRS-EE-81-1 AFOSR-TR-81-0311 NL

1 OF 2
AD
A097375

12

A STUDY OF AVIONICS TIME DIVISION MULTIPLEX BUS SIMULATION

eirs
ENGINEERING & INDUSTRIAL RESEARCH STATION
ELECTRICAL ENGINEERING

By

MALCOLM D. CALHOUN, Ph.D.

and

BEHROOZ KHATIRZAD

Prepared For:

AIR FORCE OFFICE OF SCIENTIFIC RESEARCH
BOLLING AFB, DC 20332

Mississippi State University
Mississippi State, Miss. 39762

MSSU-EIRS-EE-81-1

COLLEGE OF ENGINEERING ADMINISTRATION

WILLIE L. MCDANIEL, PH.D.
DEAN, COLLEGE OF ENGINEERING

WALTER R. CARNES, PH.D.
ASSOCIATE DEAN

RALPH E. POWE, PH.D.
ASSOCIATE DEAN

LAWRENCE J. HILL, M.S.
DIRECTOR, ENGINEERING EXTENSION

CHARLES B. CLIETT, M.S.
AEROSPACE ENGINEERING

WILLIAM R. FOX, PH.D.
AGRICULTURAL & BIOLOGICAL ENGINEERING

JOHN L. WEEKS, JR., PH.D.
CHEMICAL ENGINEERING

ROBERT M. SCHOLTES, PH.D.
CIVIL ENGINEERING

B. J. BALL, PH.D.
ELECTRICAL ENGINEERING

W. H. EUBANKS, M.ED.
ENGINEERING GRAPHICS

FRANK E. COTTON, JR., PH.D.
INDUSTRIAL ENGINEERING

C. T. CARLEY, PH.D.
MECHANICAL ENGINEERING

JOHN I. PAULK, PH.D.
NUCLEAR ENGINEERING

ELDRED W. HOUGH, PH.D.
PETROLEUM ENGINEERING

For additional copies or information
address correspondence to

ENGINEERING AND INDUSTRIAL RESEARCH STATION
DRAWER DE
MISSISSIPPI STATE UNIVERSITY
MISSISSIPPI STATE, MISSISSIPPI 39762

TELEPHONE (601) 325-2266

Mississippi State University does not discriminate on the basis of race, color, religion, national origin, sex, age, or handicap.

In conformity with Title IX of the Education Amendments of 1972 and Section 504 of the Rehabilitation Act of 1973, Dr. T. K. Martin, Vice President, 610 Allen Hall, P. O. Drawer J, Mississippi State, Mississippi 39762, office telephone number 325-3221, has been designated as the responsible employee to coordinate efforts to carry out responsibilities and make investigation of complaints relating to nondiscrimination.

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE	READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER AFOSR-TR- 81-0311 · 2. GOVT ACCESSION NO AD-A1[illegible]	3. RECIPIENT'S CATALOG NUMBER 575
4. TITLE (and Subtitle) A STUDY OF AVIONICS TIME DIVISION MULTIPLEX BUS SIMULATION	5. TYPE OF REPORT & PERIOD COVERED FINAL 1 Jan 1980 to 31 Dec 1980 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) MALCOLM D. CALHOUN and BEHROOZ KHATIRZAD	8. CONTRACT OR GRANT NUMBER(s) AFOSR-80-0126
9. PERFORMING ORGANIZATION NAME AND ADDRESS Mississippi State University Drawer EE Mississippi State, MS 39762	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 2304/D9 61102F
11. CONTROLLING OFFICE NAME AND ADDRESS Directorate of Mathematical and Information Sciences, Lt. Col. George W. McKemie, AFOSR/NM Bolling AFB, DC 20332	12. REPORT DATE DEC. 1980 13. NUMBER OF PAGES 156
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)	15. SECURITY CLASS. (of this report) Unclassified 15a. DECLASSIFICATION DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release;
distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

SIMULATION
TIME DIVISION MULTIPLEXING
MUXSIM
AVIONICS DATA BUS

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

Utilization of AFAL's MUXSIM Simulation Program is made, linking MUXDA and MUXDB to GASP IV. Computer Simulations are made to compare FORTRAN, GASP IV, GPSS II, SIMSCRIPT II. ADA and ECSS II are considered as possible simulation tools.

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

Mississippi State University
Department of Electrical Engineering

A STUDY OF AVIONICS TIME DIVISION MULTIPLEX BUS SIMULATION.

For Period Covering
1 January 1980 to 31 December 1980

Final Report
AFOSR-80-0126

Principal Investigator
Malcolm D. Calhoun, Ph.D.

Report Prepared By:
Behrooz Khatirzad

Prepared For:

Air Force Office of Scientific Research
Bolling AFB, DC 20332

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

TABLE OF CONTENTS (Continued)

TABLE OF CONTENTS (Continued)

TABLE OF CONTENTS (Continued)

LIST OF FIGURES

LIST OF FIGURES (Continued)

LIST OF FIGURES (Continued)

LIST OF TABLES

CHAPTER I

INTRODUCTION

Progress in digital technology has led to the development of shared information among electronic subsystems. For example, a number of digital processors may be interconnected via one common communication channel. Recent advances in microprocessors have made distributed processing a reality in many practical applications: manufacturing, computing, integrated avionics systems, etc. One technique for transferring information between several devices on one common channel is known as time division multiplexing; the channel over which the information is transferred is called a multiplex data bus. [1]

The multiplex system is a collection of electronic devices which send or receive signals for encoding and/or decoding; also, the system is capable of storing for future dispersal messages which arrive simultaneously. The components of the multiplex system are, (1) the bus controller, (2) remote terminals, (3) subsystems which may have embedded remote terminals, and (4) the data bus. See Figure 1-1. The data bus conveys information between the bus controller and the remote terminals (RT). The number of RT's on the data bus depends on the complexity of the desired system. The bus controller initiates information transfers on the data bus and is an integral part of the multiplex system. A subsystem is a functional unit which receives data transfer service from the data bus. Frequently it is necessary to have more than one data bus; a data bus which has more than one path between the subsystems is called a redundant data bus. Figure 1-1 illustrates a redundant data bus architecture.

Discrete event simulation is a viable means of modeling digital multiplex systems. Questions concerning data bus utilization and/or bus traffic loading should be answered prior to the hardware development of the system. A simulation model may be used in the preliminary design and accuracy testing of a digital multiplex system. One such simulation model which has been developed for this purpose is the Multiplex System Simulator (MUXSIM) [2] . In building the model for MUXSIM, FORTRAN IV and the GASP IV simulation languages were used; however, other simulation languages such as GPSS II or SIMSCRIPT II may be used. In selecting the simulation language to use in modeling a multiplex system, care should be taken that the simulation language matches the host computer system.

An overview of several current simulation languages is presented in Chapter II. In Chapter III, the general views of MUXSIM and the analysis of the dynamic part of the MUXSIM system are described in detail. The purpose of the dynamic MUXSIM (MUXDA and MUXDB) is to model the time-variant or stochastic aspects of the system and to obtain the simulation results on such system parameters as queue size, time delay, system failure, etc. In Chapter IV, a programming example of a single queue, single bus is presented in FORTRAN IV, and three different simulation languages, (GASP IV, SIMSCRIPT II, and GPSS II). Chapter V is a comparative study of the simulation languages based on the results of Chapter IV. Recommendations regarding the choice of a language for multiplex simulation are included.

All programs and modified programs which are used in this report are included in the Appendix. Numbers enclosed in brackets [ ] refer to the reference list at the end of the report.

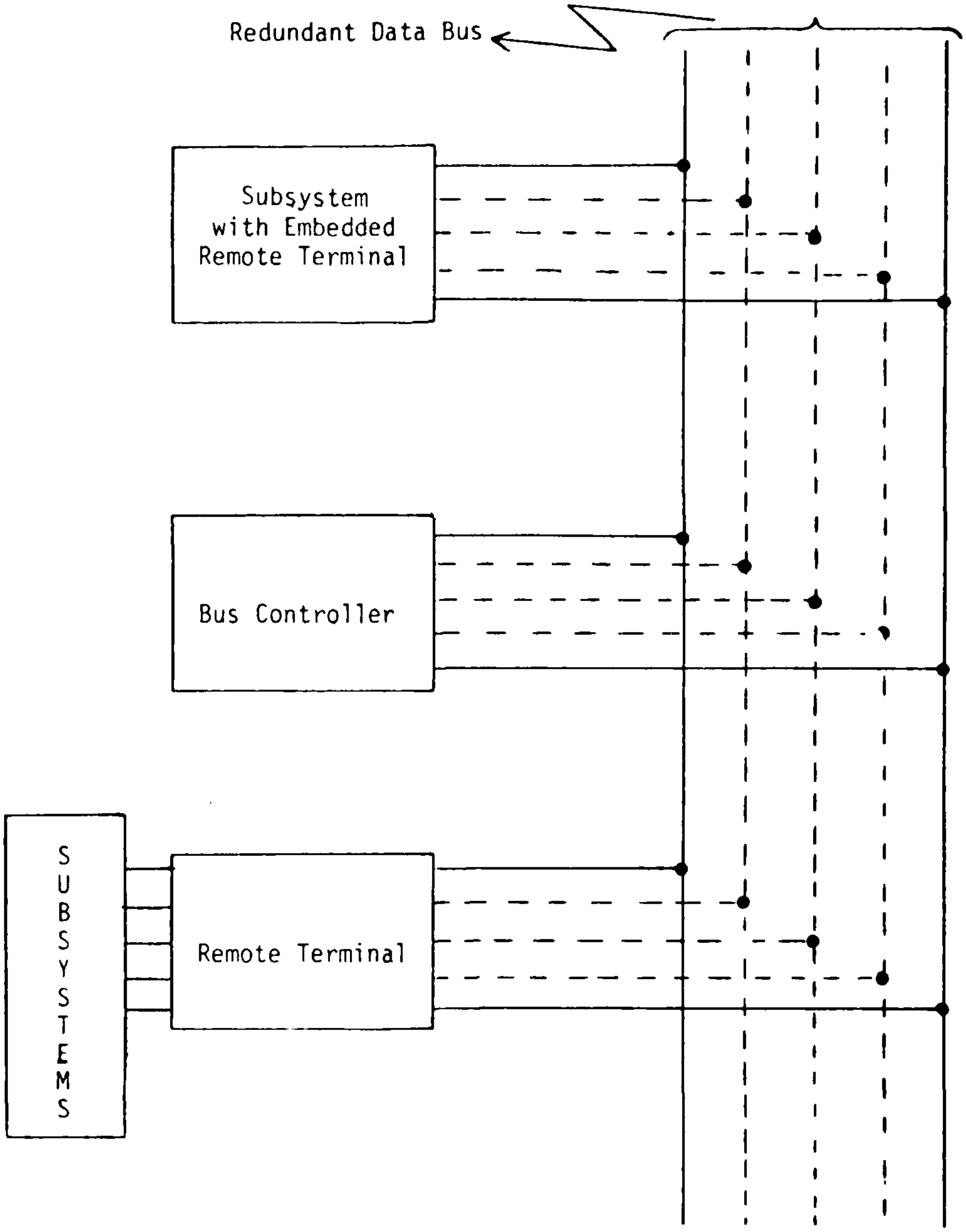


Figure 1-1 Sample Multiplex Bus Architecture.

CHAPTER II

SIMULATION LANGUAGE

Simulation is a good approach to analysis in the design and operation of a complex system. It is necessary for the modern engineer to be familiar with the techniques of simulation.

Large scale system modeling, using similation is very dependent on the digital computer; therefore, one who is interested in simulation modeling should have a basic knowledge of computer science. In this project, it is necessary to know FORTRAN IV as a requirement for learning and working with the GASP IV simulation language.

The definition of simulation: Simulation is the process of designing a model of a real system and conducting experiments with this model for the purpose of either understanding the behavior of the system or evaluating various strategies (within the limits imposed by criterion or set of criteria) for the operation of the system [3].

In simulation modeling, the engineer seeks to describe a system and its behavior. For this purpose, theories or hypothesis must be constructed. These theories are used to predict future events.

The greatest advantage of simulation is its powerful education and training application, because the development and use of simulation allows a means to find the problems which may happen in the real world; this, in turn, helps in understanding and learning how to handle the difficulties or problems. It should be pointed out that the development of a good simulation model may require a lot of time and expense.

In addition, simulation modeling is a type of art work, and therefore requires a talented engineer. Furthermore, the complexity of the system may be such that it is not amenable to simulation. Thus, it is possible that the results of a simulation do not always fit in the real world. Another reason for the difference between simulation results and real world is that we cannot create all the conditions in our simulation model; therefore, when possible, the results of a simulation should be compared with the direct experiment in the real life systems. If there is too much difference in the results, the model should be modified to overcome many difficulties in obtaining a good match between the model and actual conditions.

Is it always possible to perform a direct experiment? The answer is not always yes, because the direct experiment may be too costly and time consuming, or it may be too difficult to maintain the same condition for each run of the experiment. Also, it may not be possible to create many types of alternatives in the real life. Furthermore, the simulation result is numerical. These numbers may be truncated several times during the simulation process itself; therefore, there is always danger of obtaining an incorrect result or a slightly different result from the real world.

GENERAL VIEWS OF SIMSCRIPT II

SIMSCRIPT II is a very impressive and flexible computer programming language. It can be used for general programming problems [4] [5]. SIMSCRIPT II is divided into five language levels, which are as follows:

1. Level One or Elementary User's Language

This is designed to introduce programming concepts if one is not familiar with computer programming. This level is a simple teaching language.

2. Level Two or Level of FORTRAN

This is almost like the FORTRAN language, but is different in specific features. For example, all variables are not real unless otherwise defined; SQRT (square root) and other FORTRAN functions are not allowed to be used as variable names.

3. Level Three or Level of PL/I or ALGOL

This level is almost comparable to PL/I or ALGOL, but as in level two, they have many differences.

4. Level Four or Entity-Attribute-Set-Level

This level contains the entity-attribute-set of this language. The simulation program in this level should have a preamble, and every statement which appears in the preamble should define the existence of a class of entities. An entity can belong to other entities, have sets of other entities, and may have attributes.

5. Level Five or Simulation-Oriented Feature

Levels one through four present a general programming language, but level five is different, in that it provides concepts and programming features for discrete-event simulation. Discrete-event simulation handles models whose entities interact with one another at discrete times, instead of continuously. This level deals with concepts and statements made to help in modeling systems.

General Structure of Simulation Programming in SIMSCRIPT II

Every EVENT must be defined in the preamble, scheduled by the modeler, and must be supported by an event routine. For example, for simulation of an arrival and departure, one should define arrival and departure in the preamble. In the main program, the arrival should be scheduled and after it, an event arrival must be written. A schedule of departures should be in the event arrival and departures should be supported by an event departure. Figure 2-1 illustrates the general layout of the above example.

```
PREAMBLE
   EVENT NOTICES INCLUDE ARRIVAL AND DEPARTURE
END
MAIN
   SCHEDULE AN ARRIVAL
.
.
.
.
EVENT ARRIVAL
      .
      .
      .
      .
      SCHEDULE A DEPARTURE
      .
      .
      .
      .
      RETURN
END
      EVENT DEPARTURE
      .
      .
      .
      .
      RETURN
END
```

Figure 2-1 General Layout for the Simulation of an Arrival and Departure.

GENERAL VIEWS OF GASP IV

GASP IV was developed by Dr. A. Alan B. Pritsker at Purdue University and is based on GASP II, which was developed at Arizona State University, which in turn was based on the original GASP developed at U. S. Steel by Mr. Phillip J. Kiviat. [6]

GASP is an acronym for General Activity Simulation program. GASP IV is a specialized language for constructing simulation models of computer systems. It is a powerful and a well-documented simulation language. GASP IV is a FORTRAN based simulation language and does not require a separate compiling system. It is easy to maintain on any machine which has a FORTRAN IV compiler. This simulation language is used for discrete, continuous, and combined discrete/continuous modeling and is the only simulation language with this capability. It is easy to modify and extend to meet the needs of particular applications. GASP IV has the capabilities for event control, to update system variables, to initialize the state of system, and to collect the statistical value.

A simulation program written in GASP IV is divided into two parts, a user part and a GASP IV part. The user part consists of the main program and subroutine. In the main program, all non-GASP variables that remain constant for all simulation runs should be initialized. Some of the most used GASP subroutines are described below:
Subroutine GASP is called via the main program. The general layout of the main program is shown in Figure 2-2.
Subroutine INTCL is used to initialize non-GASP variables at the start of each run.

```
C  MAIN PROGRAM
   DIMENSION NSET (NNSET)
C  NNSET to be specified
   COMMON (GASP VARIABLES)
   COMMON (NON-GASP VARIABLES)
   EQUIVALENCE (NSET (1), QSET (1))
C  Initialization of non-GASP variables
C  Initialization of Card Reader Value, NCRDR and Printer Value,
C  NPRNT.
   CALL GASP
C  If more runs are desired, insert GO TO statement to either
C  reinitialize non-GASP variables or to CALL GASP again.
   STOP
   END
```

Figure 2-2 The General Layout of the Main Program for GASP IV.

The user written subroutine EVNTS (IX) is called to pick up the event code IX and to call the appropriate event code. The general form event code is given in Figure 2-3.

```
    SUBROUTINE EVNTS (IX)
    DIMENSION NSET (1)
    COMMON QSET (1)
    COMMON (GASP VARIABLES)
    COMMON (NON-GASP VARIABLES)
    EQUIVALENCE (NSET (1), QSET (1))
C   For Single Queue and Single Server
C   Simulation program which is written in this thesis
C   IX has been specified as follows:
C   If IX is 20, Event Arrival will occur.
C   If IX is 30, Event Begin Service will occur.
C   If IX is 40, Event Finish Service will occur.
    GO TO (20, 30, 40), IX
20  CALL ARR
    RETURN
30  CALL BEGS
    RETURN
40 CALL FINS
   RETURN
   END
```

Figure 2-3 Layout of Subroutine EVNTS.

Subroutine OTPUT produces some information in addition to the standard GASP summary report. It can be used as an end of simulation event.

The GASP part of the simulation program consists of the subprogram that prepares for the following functions: data collection, statistics computation and reporting, monitoring and error reporting, random deviate generation, data storage and retrieval data, and event initialization and mode controller. Figure 2-4 shows the flow chart of the GASP program.

Figure 2-5 presents a diagram showing the relationship of the GASP IV subprograms and the user written subprograms. The lines in Figure 2-5 represent one subprogram calling another. Each of the user written subprograms can call any of the GASP IV subprograms. Lines indicating such calls are problem specific and are not shown in the figure. Subprogram names, having both a solid box and a dashed box around them, are usually written by the user. GASP IV gives a "dummy" version if no user version is written.

GASP Input Data Cards

The GASP program has standard input data cards besides the user inr Jt cards. The user input data card is placed before or after, or both before and after the GASP input data card, depending on the type of program.

There are twelve types of input data cards as described below:

Data Card Type I

Data card type I is used for recording the name of the programmer, the number of the project, the date and number of the simulation run.

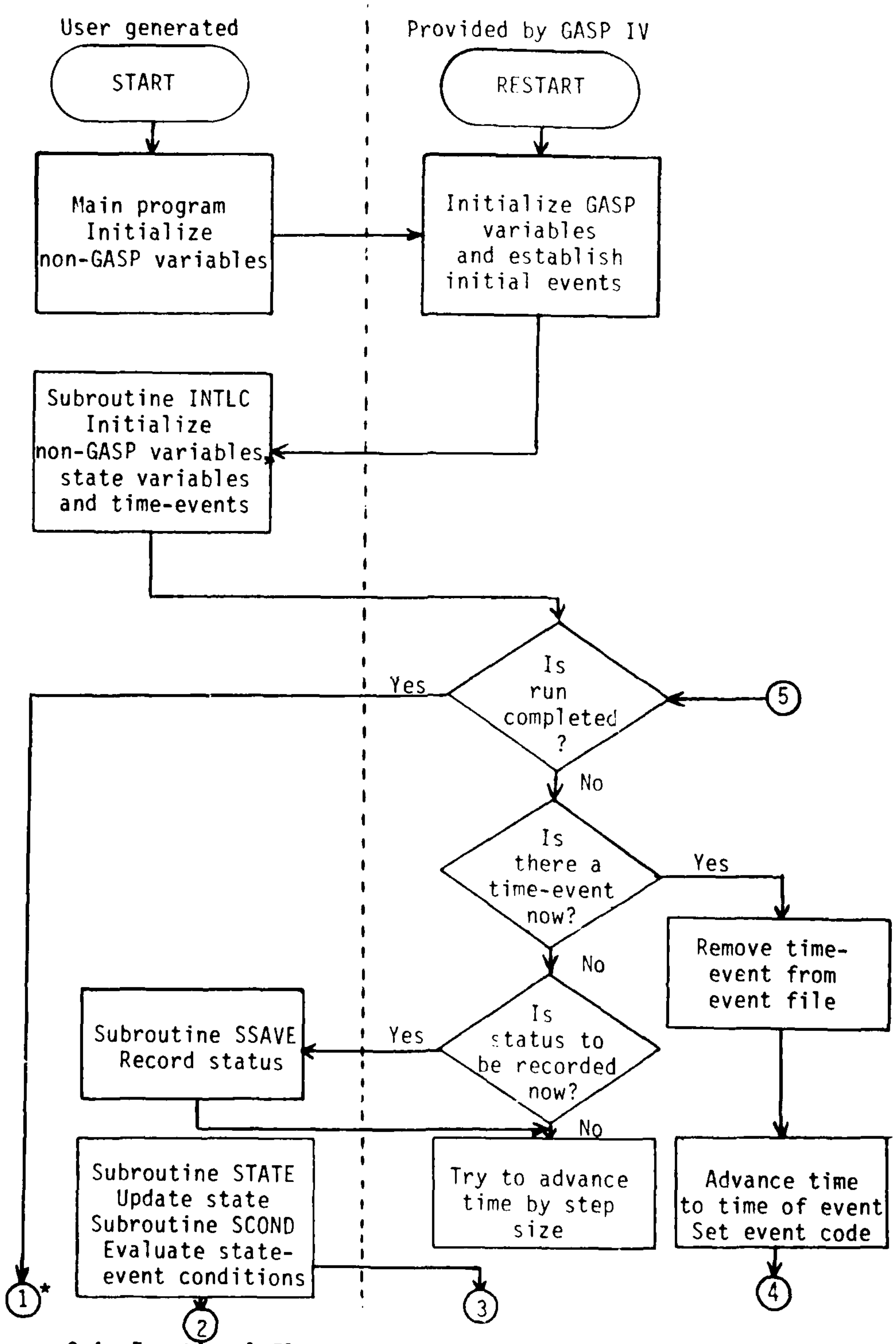

User generated
Provided by GASP IV
START
RESTART
Main program Initialize non-GASP variables
Initialize GASP variables and establish initial events
Subroutine INTLC Initialize non-GASP variables, state variables and time-events
Is run completed ?
Yes
No
5
Is there a time-event now?
Yes
No
Remove time-event from event file
Is status to be recorded now?
Yes
No
Subroutine SSAVE Record status
Subroutine STATE Update state Subroutine SCOND Evaluate state-event conditions
Try to advance time by step size
Advance time to time of event Set event code
1*
2
3
4

Figure 2-4 Functional Flow Chart of a GASP IV Program,

* Numbered circles refer to destination points in the program.

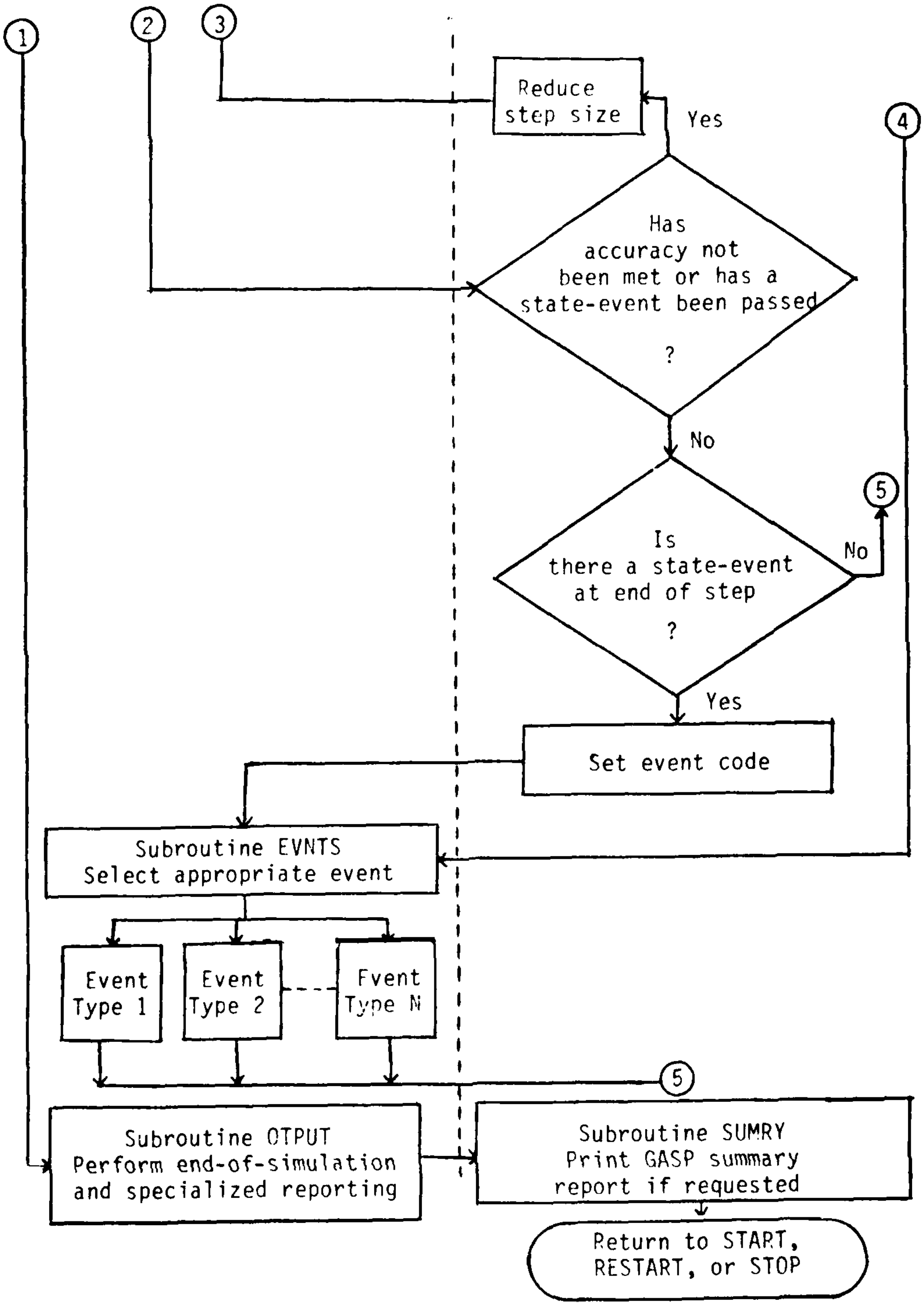

1
2
3
Reduce step size
Yes
4
Has accuracy not been met or has a state-event been passed ?
No
5
Is there a state-event at end of step ?
No
Yes
Set event code
Subroutine EVNTS Select appropriate event
Event Type 1
Event Type 2
Fvent Type N
5
Subroutine OTPUT Perform end-of-simulation and specialized reporting
Subroutine SUMRY Print GASP summary report if requested
Return to START, RESTART, or STOP

Figure 2-4 Functional Flow Chart of a GASP IV Program (Continued)

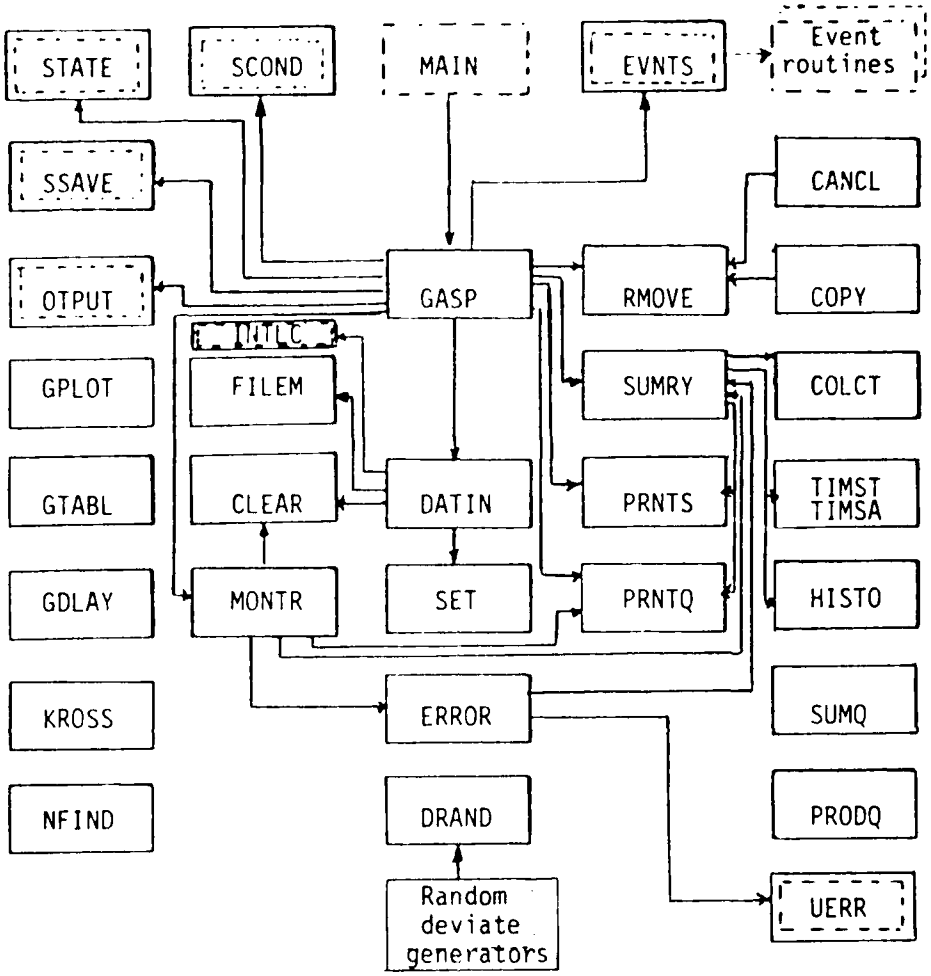


Figure 2-5 Relation of GASP IV and user subprograms. [6]

□ GASP IV subprogram

⧈ Dummy subprogram

⬚ User programs required

A → B Subprogram A calls Subprogram B

The format is (3A4, 3X, 5I5, 15I1) for type I data cards.

Data Card Type II

Data card type II is used if the control variable (LLSUP (2)) is less than one. It consists of information about subroutine COLCT, TIMST, and TIMSA, number of histograms, number of parameter sets, number of plots or tables, number of random streams, maximum allowable number of entries in the file storgae area (NSET/QSET), maximum number of attributes per entry in NSET/QSET, number of files in NSET/QSET, dimension of NSET/QSET, number of derivative equations, number of equations defining in state level and number of state condition flags (LFLAG) employed. The format is (15I5).

Data Card Type III

Input card type III is used only if the number of sets of statistics collected by subprogram COLCT (NNCLT) is greater than zero and the control variable (LLSUP (3)) is less than one. The format of this card is (5X, I5, 2A4) and it contains labels associated with variables used in COLCT.

Data Card Type IV

Input data card type IV is used only if the number of sets of statistics collected by subprograms TIMST and TIMSA is greater than zero and the control variable (LLSUP (4)) is less than one. The format of this card is (5X, I5, 2A4, E10.0) and it consists of a label for TIMST, TIMSA and the initial value for the time persistent variable.

Data Card Type V

This input data card is used only if the number of histograms (NNHIS) is greater than zero and the control variable (LLSUP (5)) is less than one. The format of this type of card is (5X, I5, 2A4, 7X, I5, 2E10.0). It consists of a label of histogram, number of cells of each histogram, upper limit of the first cell of the histogram, and the width of a cell for histogram.

Data Card Type VI

Data input card type VI is divided into two types, Type A and Type B.

Type A is used only if the number of plots and/or tables (NNPLT) is greater than zero and the control variable (LLSUP (6)) is less than one. The format used is (5X, I5, 2A4, 7X, 3I5, E10.0). This card gives information about the label of plot, index of tape, number of variables for the table or plot, keys for specifying the type of table or plot, and intervals between successive plot points.

Type B is used only if IJ (Index) is less than the number of variables to be plotted or tabled. The format used is (5X, I5, A1, 2A4, 1X, 2I5, 2E10.0). This card gives plot symbols, labels for plots, keys for specifying lower and upper limits and values associated with lower and upper limits of plot ordinates.

Data Card Type VII

This type of card is used only if the number of files in the file storage area (NSET) is greater than zero and the control variable (LLSUP (7)) is less than one. The format is (14I5). This card gives information about ranking attributes for files.

Data Card Type VIII

This type of card is used only if the number of files (MNFIL) in the file storage area (NSET) is greater than zero and the control variable (LLSUP (8)) is less than one. The format of this card is

(1415). This card consists of keys for priority systems to be used in the files.

Data Card Type IX

This type of card is used only if the number of state and derivative equasions (NNEQT) is greater than zero and the control variable (LLSUP (9)) is less than one. The format of this card is (2I5, 5E10.0). This card has information about accuracy, mimimum and maximum step size permitted, and keys between communication points.

Data Card Type X

This type of card is used only if the number of parameter sets (NNPRM) is greater than zero and the control variable (LLSUP (10)) is less than one. The format of this card is (5X, I5, 4E10.0). This data card has information about parameter set numbers and parameter numbers.

Data Card Type XI

Data card type XI is used only if the control variable (LLSUP (11)) is less than one. The format of this card is (4I5, 2E10.0, I5, (6I5)). This data card yields information about stopping the simulation, whether statistical array should be cleared during initialization, and the initialization of the random number seed.

Data Card Type XII

This type of card is used only if the number of files in NSET (NNFIL) is greater than zero and the control variable (LLSUP (12)) is less than one. The format is (5X, I5, (6E10.0)). This data card gives information about the file number for attributes. If this number is equal to zero, it shows the end of data card type XII.

Data Card Type 0

Data card type 0 is used only if the remaining number of runs (NNRNS) is greater than one and the indicator used in DATIN for initialization (IICRD) is equal to zero. The format of this card is (15I1, I5). This card is used for multiple runs and it has a different value for different programs.

GENERAL VIEWS OF GPSS II [7]

GPSS II is an acronym of General Purpose Systems Simulator II. This program language is one of the easiest simulation languages. It is not as flexible as GASP IV or other advanced simulation languages, because of a limitation on the number of blocks, storage areas, and transaction in systems.

In order for a system to be simulated, it must be reducible to a series of operations performed on units of traffic. The units of traffic upon which the system operates relies on the nature of the system. Traffic may be work items in a production line, electrical pulses in a digital circuit, or messages in a communications system. Transactions are the units of traffic that are made and used by the simulator. These transactions have certain properties which conform to characteristics of traffic in a system. Each transaction is associated with a priority, which is one integer between zero and seven. When competing for service, the transaction with the numerically higher priority will be the first to be processed. If transactions have the same priority, the one that has been delayed longer will be selected first.

The program supplies block types, which present operations in a model of the system equivalent to the actions happening in the real system. Each block specifies a number of clock units that the transactions are to spend in that block. The number may be made to depend upon a number of factors within the system itself, or it may be constant or computed from statistical distribution. Every block specifies the next step to which a transaction will be sent when its computed time interval is completed.

The user gives each block an identifying number to designate the path of flow. The fundamental properties of some of these block types are described and all the operations which may be performed are discussed. Before the properties of these blocks can be discussed, it is necessary to specify the format of the block types.

The card fields of the GPSS II blocks are as follows:

Field	Columns
LOCATION	2-6
NAME	7-18
X	19-24
Y	25-30
Z	31-36
SELECTION MODE	37-42
NEXT BLOCK A	43-48
NEXT BLOCK B	49-54
MEAN TIME	55-60
MODIFIER	61-66
COMMENTS	67-80

In these fields, all numbers should be left justified.

Description of GPSS II Blocks

ADVANCE

Purpose: In this block, transaction waits while the clock advances.

Operand: This block has no operand.

Condition for entry acceptance: It does not refuse entry under any condition.

ASSIGN

Purpose: This block modifies the value of the parameter.

Operand: X field gives the parameter numbers. Y field specifies system variables.

Condition for entry acceptance: It never refuses entry.

ASSEMBLY

Purpose: This block is used to terminate the number of assembly sets (assembly sets are original and duplicated blocks).

Operand: X field specifies the assembly count, which must be at least two.

Condition for entry acceptance: It always accepts entry.

COMPARE

Purpose: This block tests the relationship between two system variables.

Operand: X field gives the system variable that is going to be tested. Y field is specified by Mnemonics (kind of relationship). Z field gives the system variable that is going to be tested.

Condition for entry acceptance: It refuses entry whenever the relationship is false.

ENTER

Purpose: This block places one or more units into storage.

Operand: X field specifies the storage number. Y field specifies the number of units to be stored.

Condition for entry acceptance: It never refuses entry.

GENERATE

Purpose: The GENERATE block creates transaction.

Operand: X field gives the time of the first transaction. Y field specifies the limit count. Z field gives the priority of a central transaction. Mean time should be given for the interval between the creation of two transactions. The modifier can be specified two ways, first, by a constant value which gives a uniform random variable. Second, by a function which gives the form of distribution.

Condition for entry acceptance: It never accepts entry.

GATE

Purpose: This block tests the status of some entities.

Operand: X field is given by Nmemonic, followed by a number (facility number).

Condition for entry acceptance: It refuses entry whenever indicated status is false.

INDEX

Purpose: The INDEX block is used to compute a transaction parameter value for temporary use. A constant is added to the specified parameter and stores the result in Parameter 1.

Operand: X field gives a parameter number. Y field gives the constant value.

Condition for entry acceptance: It never refuses entry.

LOGIC *

Purpose: This block changes the status of the LOGIC switch.

Operand: X field is specified by S(set) or R(reset), plus number.

Condition for entry acceptance: It never refuses entry.

* Initially LOGIC switches are reset (0).

LEAVE

Purpose: The LEAVE block takes a unit out of storage.

Operand: X field specifies the storage number. Y field gives the number of units to be taken out.

Condition for entry acceptance: It never refuses entry.

LOOP

Purpose: This block causes a transaction to cycle through a set of blocks several times.

Operand: X field gives a parameter number.

Condition for entry acceptance: It never refuses entry.

MARK

Purpose: This block marks the transaction with the current clock time.

Operand: X field is either blank or contains a parameter number. When using a parameter number, the parameter number should be marked.

Condition for entry acceptance: It never refuses entry.

MATCH

Purpose: The MATCH block is used to synchronize the movement of the assembly set.

Operand: X field specifies the block number of a MATCH block.

Condition for entry acceptance: It always accepts entry.

PREEMPT

Purpose: This block attempts the higher level usage of facility.

Operand: X field gives the facility number.

Condition for entry acceptance: It refuses entry if the facility has already been preempted.

PRIORITY

Purpose: This block is used to set PRIORITY of entry transactions.

Operand: X field is used to specify the value of PRIORITY. Y field is either blank or buffer.

Condition for entry acceptance: It never refuses entry.

PRINT

Purpose: This block is used to print out some expected value.

Operand: X field specifies the first location of SAVEX to be printed. Y field specifies the last location of SAVEX to be printed.

Condition for entry acceptance: It never refuses entry.

QUEUE

Purpose: It records one or more entries into a Queue.

Operand: X field gives the QUEUE number. Y field specifies the number to be added into QUEUE.

Condition for entry acceptance: It never refuses entry.

RELEASE

Purpose: The RELEASE block is used to end service on facility.

Operand: X field specifies the number of facility.

Condition for entry acceptance: It always refuses entry.

RETURN

Purpose: This block is used to end the preemption.

Operand: X field gives the facility number.

Condition for entry acceptance: It never refuses entry.

SAVEX

Purpose: This block allows the user to gather and print information from the block diagram, and transmit information from one transaction to another.

Operand: X field of this block specifies a SAVEX storage location. Y field gives the system variable to be used in the modification.

Condition for entry acceptance: It always accepts entry.

SEIZE

Purpose: The SEIZE block begins service on a facility.

Operand : X field specifies the number of facility.

Condition for entry acceptance: It refuses entry if it has already seized.

SPLIT

Purpose: This block creates a duplication of each transaction that enters the block.

Operand: There is no operand.

Condition for entry acceptance: It always accepts entry.

TABULATE

Purpose: This block gives statistics on the simulation program.

Operand: X field specifies the number of TABULATE blocks.

Condition for entry acceptance: It always accepts entry.

TERMINATE

Purpose: The TERMINATE block removes transactions from the block diagram.

Operand: X field should be left blank or specified by the letter R. If R is used in the X field, the termination counter would be reduced by one.

Condition for entry acceptance: It does not refuse entry under any condition.

The standard upper limits of blocks, storage, etc. are given for GPSS II in Table 2-1:

Item	Std. Max.	Words/Item
Blocks	800	5
Facilities	200	6
Storage	200	7
Queues	200	6
Logic Switches	500	1
Savex Locations	500	1
Functions	100	4
Tables and QTABLES	100	10
Variable Statements	50	1
Transactions in System	1000	9

Table 2-1 GPSS II Standard Block Limits.

GENERAL VIEWS OF THE ADA LANGUAGE

The ADA language was designed by a team led by Jean D. Ichbiah. It has been chosen as the name for the common language, honoring Ada Augusta, the daughter of the poet, Lord Byron, and Babbage's programmer [8].

The ADA language was designed with three concerns in mind: a recognition of the importance of program dependability and maintenance, a concern for programming as a human activity, and efficiency.

The ADA language is a modern algorithmic language which contains the usual control structures, and is able to define types* and subprograms. ADA is also capable of serving the need for modularity, whereby data, types, and subprograms can be packaged. Any program in the ADA language is a series of higher level program units, with the capability of compiling separately.

The program units can be a subprogram which is an executable algorithm. Package modules are collections of entities or task modules which are concurrent calculations. The subprogram, package modules, and task modules are described in more detail as follows:

Subprogram

A subprogram is an executable unit which is the basic unit for expressing an algorithm. A subprogram can have parameters, which

* A type characterizes a set of values and a set of operations that apply to those values.

show its connections to other program units. There are two kinds of subprograms in the ADA language: (1) procedures and (2) functions. These are described below:

(1) Procedure Subprogram

A procedure subprogram is the logical equivalent to a series of actions. For example, it may read in data, update variables, or produce some output.

(2) Function Subprogram

A function subprogram is the logical equivalent to a mathematical function for computing a value.

Package Modules

A package module is composed of fundamental units to define a collection of logically related entities.

Task Modules

A task module is almost like a package module, but with more capability for parallel processing. A task can be carried out on multiple processors or with interleaved execution on a single processor, the same as procedure entries can have a parameter showing the transmission of data between tasks.

Each program unit usually consists of two parts: (1) a declarative part and (2) a sequence of statements. These are described as follows:

(1) A declarative part defines the logical entities to be used in the program unit and associates names with declared entities. A name can be a variable, a constant, or a type.

(2) A sequence of statements defines the execution of the

program unit. A statement describes the type of action to be taken. An assignment statement indicates that the current value of a variable should be replaced by a new value.

Exceptional Situation in the ADA Language

In the ADA language, sometimes the computer reaches the point where normal program execution can not continue. For example, it may be needed to access the value of an uninitialized variable. To overcome this situation, the statements of a program unit can be textually followed by an exception handler describing the action to be taken if an exceptional situation arises.

GENERAL VIEWS OF ECSS II [9]

ECSS II, or Extendable Computer System Simulator II, is called "ex-two". This program language has been constructed for simulation models of a computer-based system; it is an extension of the general purpose simulation language SIMSCRIPT II, and that language is implied as a subset. It gives a wide selection of statements and data structures for describing common computer hardware structures, software operations, and work load characteristics in a natural and straight forward notation. In general, the ECSS II program includes a preamble, a system description section, a work load description, automatic event routines, and SIMSCRIPT routines. These characteristics are described below:

(1) Preamble

The preamble statement is used only if defining new global variables, functions, and entities.

(2) System Description Section

The section which describes the system of an ECSS II program defines the simulated resources in an ECSS II model. Declarative statements in this part give the name and number of every device, specify device characteristics and capabilities, show how devices are interconnected, and describe paths through which simulated data may pass.

(3) Work Load Description Section

This part of the ECSS II program defines routines which are called processes and may be used to characterize the load on a computer system's resources. The load is determined by the environment and behavior of the program that comes to be executed as a result of environmental demands.

(4) Automatic Event Routines

This section of the program is generally used for simulation monitoring and control and for representing actions that happen outside the model itself.

(5) SIMSCRIPT II Routines

SIMSCRIPT II routines, formed by a translator, consists of the initialization routine, the translation of automatic event routines, the translation of process routines, and process routines.

Processing

Producing an executable ECSS program is a three-step procedure: translation, compilation, and editing. This is illustrated in Figure 2-6.

Simulation Reports

ECSS produces three types of outputs: system display output, statistical output, and tracing output. These are described as follows:

System Display

This report is created by the SHOW SYSTEM statement. It produces the name of each device group and specifies whether it is a device or a class, and shows characteristics of the device groups and the state of model at any point during simulation.

Statistics

ECSS II can provide statistics on model performance at any particular interval. ECSS II automatically collects data on the

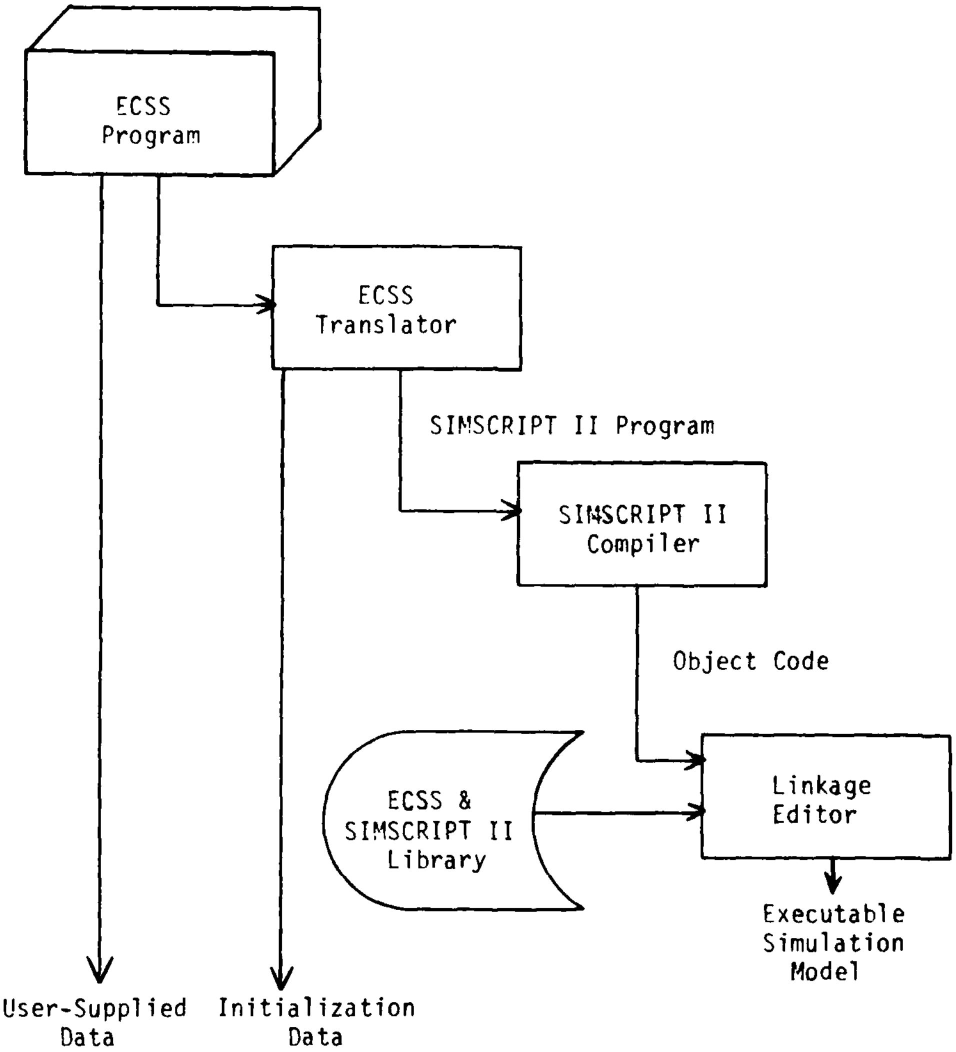


Figure 2-6 Schematic of ECSS Program Processing. [9]

activity of each device and contents of queues and is produced by the SHOW STATISTICS statement.

Tracing

This part of output gives details about interactions within a computer system and traces each step of the simulation model.

CHAPTER III

GENERAL IDEAS OF MUXSIM

MUXSIM is the abbreviation for Multiplex Simulation [2] [10] [11]. It consists of four major subsystems: the utility, the ststic, the dynamic, and the executive. These subsystems are then divided into programs, subprograms, and subroutines or modules. Most of these are written in FORTRAN and some are in GASP. The executive uses the TOPS-10 control statement. Figure 3-1 illustrates the MUXSIM System Data Flow Chart.

The four major parts of MUXSIM are described as follows:

(1) Utility Subsystem

The utility subsystem is a module of MUXSIM which manages the signal flow list, withdrawing the simulator inputs from it. This subsystem is also the management system for MUXSIM. In checking the Equipment Complement for completeness, signal deficiencies, and flagging any equipment, the utility subsystem uses the information from the signal flow list.

(2) Static Subsystem

The static subsystem deals with all the signal information, grouping, and handling, such as remote terminal assignments, word maps, message maps, as well as fixed format bus loading and utilization computation.

(3) Dynamic Subsystem

The dynamic subsystem, consisting of two discrete-event modules, handles the random messages, scheduling tasks, and computes the dynamic bus loading and time statistics. It is called "dynamic" because

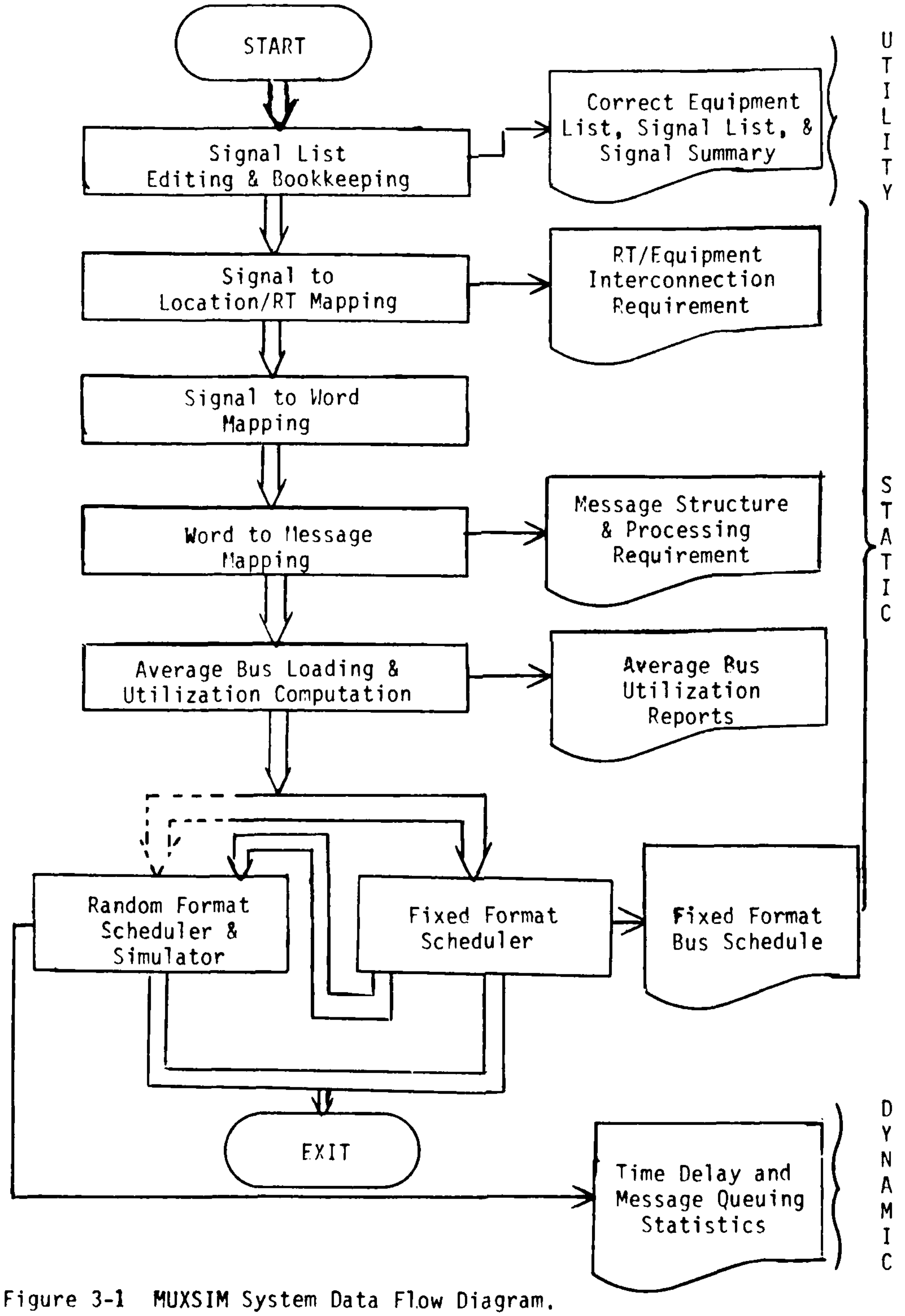

START
Signal List Editing & Bookkeeping
Correct Equipment List, Signal List, & Signal Summary
UTILITY
Signal to Location/RT Mapping
RT/Equipment Interconnection Requirement
Signal to Word Mapping
Word to Message Mapping
Message Structure & Processing Requirement
STATIC
Average Bus Loading & Utilization Computation
Average Bus Utilization Reports
Random Format Scheduler & Simulator
Fixed Format Scheduler
Fixed Format Bus Schedule
EXIT
Time Delay and Message Queuing Statistics
DYNAMIC

Figure 3-1 MUXSIM System Data Flow Diagram.

stochastic events characterizing such phenomena as multiplex system failure, bus noise, and time variable data transfer requirements are considered. This system uses the simulation language GASP IV as a component.

(4) Executive Subsystem

The executive subsystem supplies the interface between the user and the other three subsystems: the utility, the static, and the dynamic. It has an interactive section with optional coaching to assist the user and to make learning the simulator operation easier. Figure 3-2 shows the MUXSIM Modular Software Structure and their relation to one another.

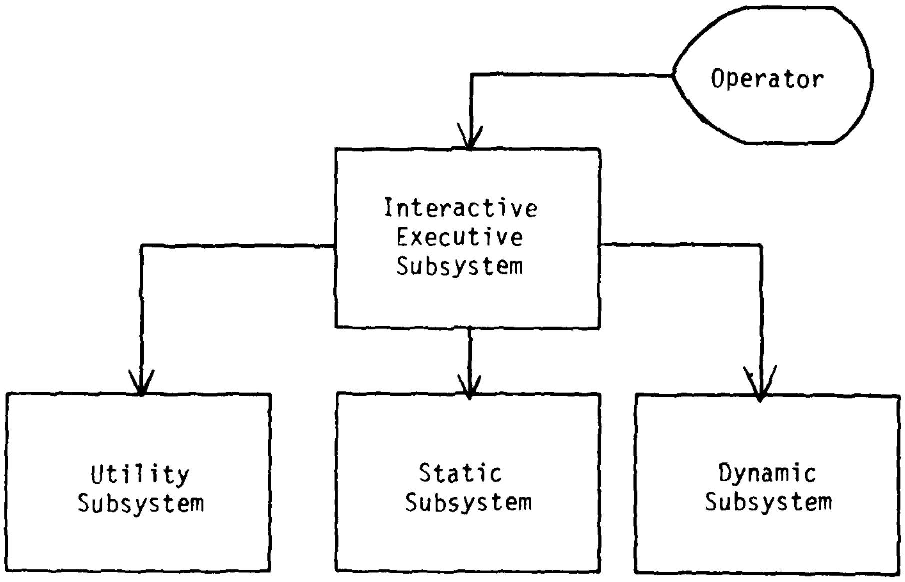


Figure 3-2 MUXSIM Modular Software Structure.

The Purpose of the MUXSIM System

The purpose of the MUXSIM system is to prepare the design and design accuracy of a digital system, used by those interested in carrying out the system design of a digital information transfer/ multiplex system. MUXSIM directs questions of data bus utilization and/or bus traffic loading.

The MUXSIM programs serve to combine specific analysis and prototype hardware. MUXSIM provides a means of interacting parts of the detailed analysis (such as updating rate requirements, sampling requirements, data buffering requirements, bus data requirements, processing delay requirements) for numerous point-to-point signaling into logical requirements which can be confirmed for compatible operation by a computer, before attempting a hardware development program.

MUXSIM is devised to be applicable to a set of multiplex system designer's questions that cannot be promptly answered by other available means.

DESCRIPTION OF THE MUXDA PROGRAM

Demand Access Transfer

This model basically deals with a demand access information transfer over the bus. The demand access messages are transmitted after the fixed message requirements are finished and until either the demand messages are used up or time has come for the start of the next fixed message transmission sequence. The central controller should know the length of every transmission and prevent the transmission of a demand message if it will interfere with the start of a fixed message.

Shown in Figure 3-3 is the demand message multiplex system schematic representation.

Statement of the Problem

MUXDA represents an information transfer system which has a fixed format data transfer foreground and an interrupt enabled demand access first-in, first-out background. This system is expected to reduce bus loading and the delay in access of the sporadic data.

Some of the assumptions made to allow the simulation to cycle faster are: error and failure-free environment, interrupt system which allows the Central Control to initiate command/response requests for the demand access data, foreground with fetch messages on a fixed telemetry format command/response basis, and foreground-background mode similar to hybrid analog-digital real-time operating system. Other assumptions are that the foreground transmission has no sporadic motion, and the computed bus load for each transfer is

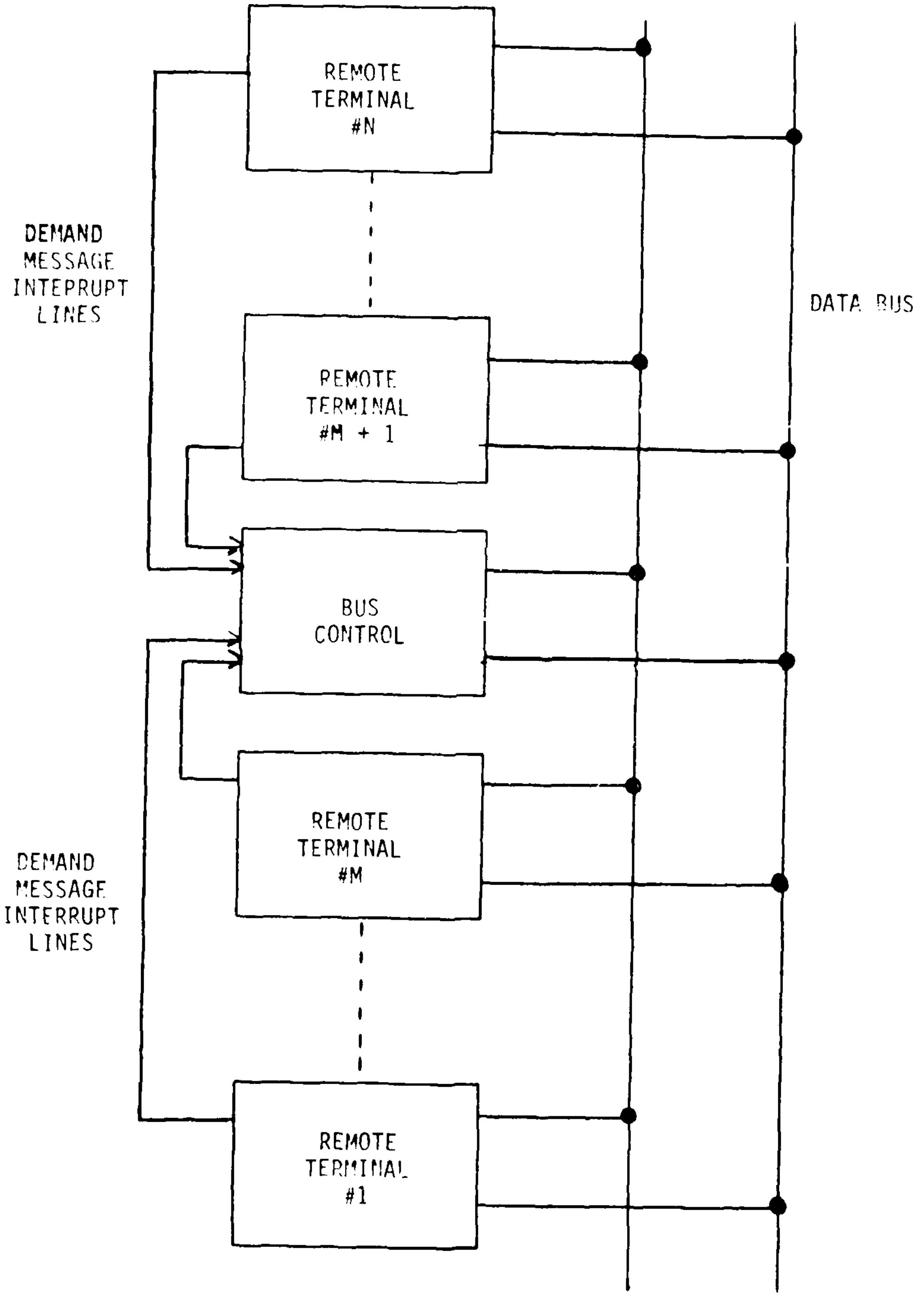


Figure 3-3 Demand Message Multiplex System Schematic Representation.

available from the static subsystem in a lumped sequence for each fundamental update interval. A further assumption is that the command response for this demand access data is initiated by central. Central knows the length of data transmission for each demand access message and does not initiate a demand message transfer which could interfere with the next foreground transmission.

Simulation Objective

The objective of this simulation is to determine bus resource utilization impact on the technique of using demand access background transfer for signals of sporadic nature.

GASP IV Simulation Structure and Program Variables for MUXDA

In this problem, there were two files used. Table 3-1 gives the definition of the files and their associated characteristics for this simulation. File 1 is the event file as per GASP IV and File 2 stores the demand message arrival. Table 3-2 defines the non-GASP variables.

Main Program, Subroutine INTLC and Subroutine EVNTS Description

Main Program

The Main Program establishes the card reader (NCRDR) and line printer (NPRNT) values and subroutine GASP is called.

In this program, a temporary disk file is not used and plot data is stored in the QSET. The MUXSIM executive system is not used, therefore it is not necessary to use subroutines CHAIN, RESTOR, and GETCOM.

ATTRIBUTES	FILE 1	FILE 2
File Definit'on	Events	Arrived demand message queue
ATRIB (1)	Event Time	Message Length
(2)	Event type = 100*I+J where I = event type J = sub type	Demand message Number
(3)	Message length (if event type is 200 otherwise it is a don't care situation)	Time of arrival in wait queue

I = Event Type

100 - FUI time start
200 - Start of FUI free time
300 - End of Demand Message
400 - Demand Message arrival

J = Sub Type

I	
100	FUI Number (1,NFUIS)
200	FUI Number (1,NFUIS)
300	Demand Message Number (1,NDM)
400	Demand Message Number (1,NDM)

Table 3-1 Definition of GASP Files.

Variable	Definition
DM (I,J)	Demand message parameter definition J = Demand message number (1,NDM) I = Demand message parameter, where: 1 = mean time between demand message occurrence 2 = Maximum Δ of uniform distribution about mean 3 = Demand message length
DMSENT (I)	Sum of demand message lengths sent for a particular FUI, where I is the FUI number I = (1,NFUIS)
FUI	Fundamental Update Interval for data transmission on bus
FUIT	The time of occurrence of the last FUI numbered 1
FUIFX (I)	Sum of the Fixed Message lengths sent for a particular FUI, where I is the FUI number I = (1,NFUIS)
FUINXT	The next FUI start time
FUIWAT (I)	Data bus idle time per FUI, computed by subtracting time of end of last Demand Message sent from start of next FUI. I is the FUI number (1,NFUIS).

MODE	MODE = 1 is FIFO;
	MODE = 2 is largest to FIT in interval first.

NDM	Maximum quantity of demand messages which the system must process.
NFUI	NFUI is the integer comment FUI number.
NFUIS	Maximum number of FUI intervals. This is the number of minor frame cycles per major frame.

Table 3-2 Definition of non-GASP Variables.

Subroutine INTLC

Subroutine INTLC is called via subroutine DATIN, in order to read in the simulation data cards and to set up the initial conditions from the input data cards or algebraic statements.

In this program, subroutine INTLC reads in Card Type I, Card Type II, and Card Type III. Card Type I defines the FUI duration and run mode, Card Type II specifies the FUI fixed message sequence duration, and Card Type III gives the demand message. All non-GASP user input data are printed out to make checking easier.

Subroutine EVNTS

Subroutine EVNTS sends control to one of the four user written subroutines: FUIT, FUIFRE, ENDDM, and DMARIV. The events of the simulation, in the order of their event code are:

100-Start of fundamental update interval time (FUIT)

200-End of fixed message transmission on bus (FUIFRE)

300-End of demand message transmission on bus (ENDDM)

400-Arrival of demand message to the queue (DMARIV)

Subroutine FUIT

Subroutine FUIT performs the following functions:

(1) In order to prevent round-off errors from accumulating and destroying the results, FUIT establishes the beginning of a major frame.

(2) In order to demonstrate the validity of operation, FUIT prints out the following for the first 100 intervals: the number of messages

in the queue, the length of demand messages transmitted in a fundamental update interval as a percent of the time available for demand message transmission, and the bus idle time as a percent of the time available for demand message transmission.

(3) For each FUI, a histogram of the length of the demand message transmission sequence is formed.

(4) By calling subroutine NXTFUI, the next FUI is scheduled.

(5) This FUI is scheduled for the start of the FUI free time.

Subroutine FUIFRE (NF)

The end of message transfer for FUI is established in subroutine FUIFRE, which performs the following functions:

(1) This subroutine processes the end of a message transmission and schedules the next.

(2) It establishes if there are any messages to be transmitted and/or the time remaining to do so.

(3) The routine tests to see if a message can be transmitted within the remaining time in FUI.

(4) This subroutine is designed to branch a specified mode and select the message for proper transmission according to that mode.

(5) FUIFRE brings the total length of demand messages up-to-date that are being sent during that FUI; also, it updates the FUI free time to the present remaining time.

Subroutine ENDDM (NM)

The start of a demand message transfer for FUI is established in subroutine ENDDM, which performs the following functions:

(1) ENDDM performs the end of the demand message arrival processing and establishes a histogram.

(2) It programs the arrival of FUIFRE while pointing out the end of the message transmission in that FUI.

Subroutine DMARIV (ND)

The arrival of a demand message on the queue is established in subroutine DMARIV, which performs the following functions:

(1) DMARIV calls for the scheduling of the next demand message arrival and processes the arrival of the demand message.

(2) It files the current demand message on the queue where it waits for transmission on the data bus.

Subroutine NXTFUI (I,WHEN)

Subroutine NXTFUI performs only one function; that is, the scheduling of the next FUI arrival.

I is the index of the FUI interval number. WHEN is the index of the next start time for this FUI interval.

Subroutine NXTDM (I)

Subroutine NXTDM performs only one function; it schedules the arrival of the next demand message.

I is the index of the Demand Message number.

Simulation Report

The MUXDA outputs are given in the following section. Figure 3-4 shows the input data echo check, provided by subroutine DATIN. Figure 3-5 presents the printout of the files that are obtained at the end of subroutine DATIN. After the GASP IV summary report, there are twenty histograms for the first run, which give the observed, the relative, and the cumulative frequencies for each cell. For example, histogram number two shows that thirty-one percent (31%) of the message has delay time less than or equal to .03 of the time unit. The length of the delay time is from the message arrival to the end of the transfer. Histogram number eleven shows that ninety five percent (95%) of the length of the demand message is less than or equal to .15 of the time unit.

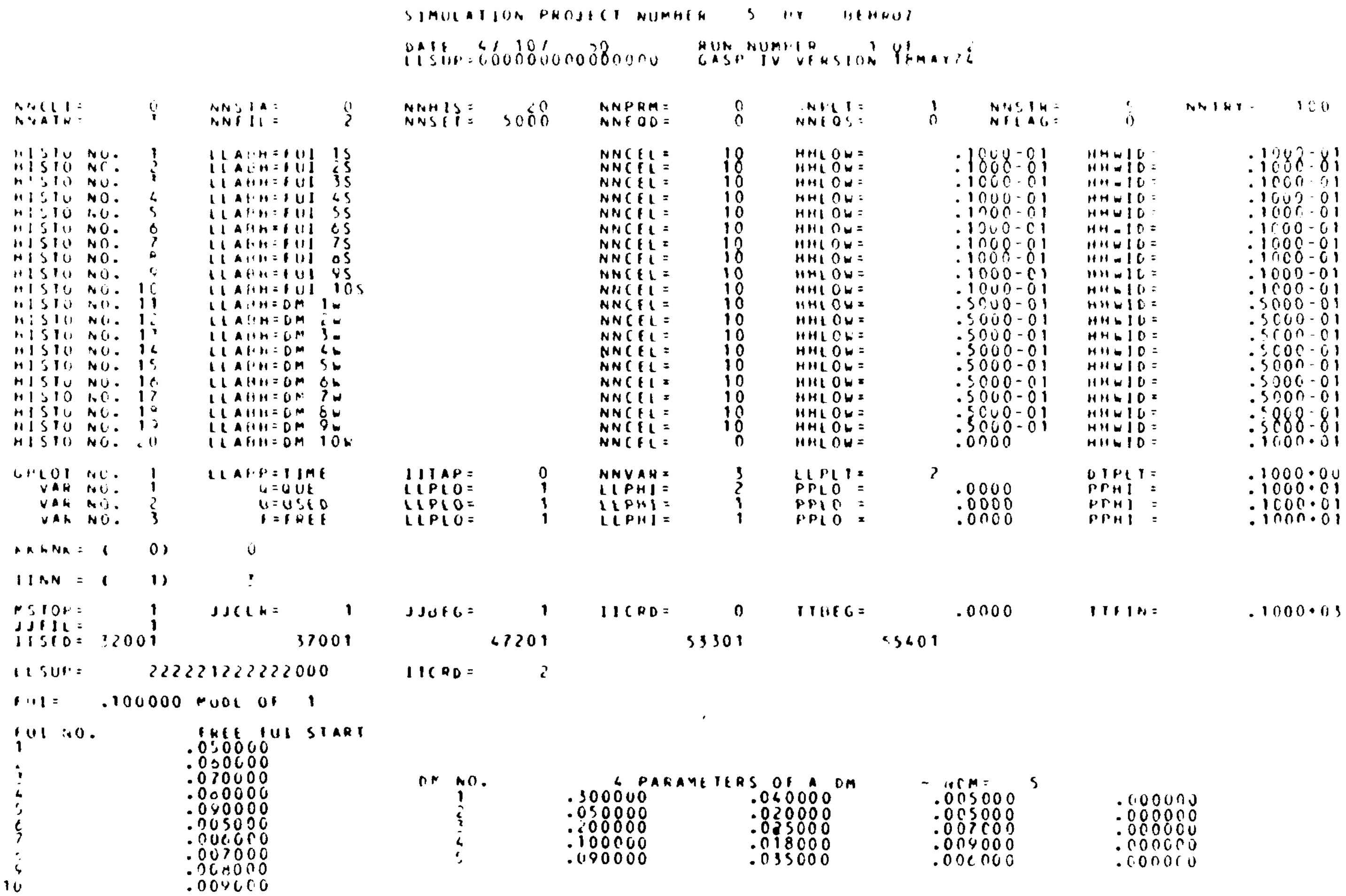
In Figure 3-6, three variables versus time are plotted. These are as follows:

(1) The number of messages in the queue, versus time

(2) The length of demand messages transmitted in a fundamental update interval as a percent of the interval time available for demand message transmission, versus time

(3) The bus idle time as a percent of the time available for demand message transmission, versus time

This plot is expanded and is shown in Figures 3-7, 3-8, and 3-9. Figure 3-10 shows the output for MUXDA in the second run. Figures 3-12, 3-13, and 3-14 are expanded plots of Figure 3-11.



```
                              SIMULATION PROJECT NUMBER    5  BY    BEHROZ

                              DATE  4/ 10/  80          RUN NUMBER    1 OF   2
                              LLSUP=6000000000000000    GASP IV VERSION 1 MAY74

NNCLT=        0     NNSTA=        0     NNHIS=   20     NNPRM=      0     NNPLT=      1    NNSTR=      5    NNTRY=   100
NNATR=        1     NNFIL=        2     NNSET=  5000    NNEQD=      0     NNEQS=      0    NFLAG=      0

HISTO NO.   1       LLABH=FUI  1S                       NNCEL=     10     HHLOW=        .1000-01     HHWID=        .1000-01
HISTO NO.   2       LLABH=FUI  2S                       NNCEL=     10     HHLOW=        .1000-01     HHWID=        .1000-01
HISTO NO.   3       LLABH=FUI  3S                       NNCEL=     10     HHLOW=        .1000-01     HHWID=        .1000-01
HISTO NO.   4       LLABH=FUI  4S                       NNCEL=     10     HHLOW=        .1000-01     HHWID=        .1000-01
HISTO NO.   5       LLABH=FUI  5S                       NNCEL=     10     HHLOW=        .1000-01     HHWID=        .1000-01
HISTO NO.   6       LLABH=FUI  6S                       NNCEL=     10     HHLOW=        .1000-01     HHWID=        .1000-01
HISTO NO.   7       LLABH=FUI  7S                       NNCEL=     10     HHLOW=        .1000-01     HHWID=        .1000-01
HISTO NO.   8       LLABH=FUI  8S                       NNCEL=     10     HHLOW=        .1000-01     HHWID=        .1000-01
HISTO NO.   9       LLABH=FUI  9S                       NNCEL=     10     HHLOW=        .1000-01     HHWID=        .1000-01
HISTO NO.  10       LLABH=FUI  10S                      NNCEL=     10     HHLOW=        .1000-01     HHWID=        .1000-01
HISTO NO.  11       LLABH=DM  1W                        NNCEL=     10     HHLOW=        .5000-01     HHWID=        .5000-01
HISTO NO.  12       LLABH=DM  2W                        NNCEL=     10     HHLOW=        .5000-01     HHWID=        .5000-01
HISTO NO.  13       LLABH=DM  3W                        NNCEL=     10     HHLOW=        .5000-01     HHWID=        .5000-01
HISTO NO.  14       LLABH=DM  4W                        NNCEL=     10     HHLOW=        .5000-01     HHWID=        .5000-01
HISTO NO.  15       LLABH=DM  5W                        NNCEL=     10     HHLOW=        .5000-01     HHWID=        .5000-01
HISTO NO.  16       LLABH=DM  6W                        NNCEL=     10     HHLOW=        .5000-01     HHWID=        .5000-01
HISTO NO.  17       LLABH=DM  7W                        NNCEL=     10     HHLOW=        .5000-01     HHWID=        .5000-01
HISTO NO.  18       LLABH=DM  8W                        NNCEL=     10     HHLOW=        .5000-01     HHWID=        .5000-01
HISTO NO.  19       LLABH=DM  9W                        NNCEL=     10     HHLOW=        .5000-01     HHWID=        .5000-01
HISTO NO.  20       LLABH=DM  10W                       NNCEL=      0     HHLOW=        .0000        HHWID=        .1000+01

GPLOT NO.   1       LLAPP=TIME          IITAP=      0   NNVAR=      3     LLPLT=      2                DTPLT=        .1000+00
  VAR NO.   1             Q=QUE         LLPLO=      1   LLPHI=      2     PPLO  =        .0000         PPHI  =       .1000+01
  VAR NO.   2             U=USED        LLPLO=      1   LLPHI=      1     PPLO  =        .0000         PPHI  =       .1000+01
  VAR NO.   3             F=FREE        LLPLO=      1   LLPHI=      1     PPLO  =        .0000         PPHI  =       .1000+01

KKRNK= (     0)      0

IINN = (     1)      7

MSTOP=      1       JJCLR=      1       JJBEG=      1   IICRD=      0     TTBEG=         .0000         TTFIN=        .1000+03
JJFIL=      1
IISED= 32001              37001              47201             53301             55401

LLSUP=        2222212222222000      IICRD=      2

FUI=     .100000 POOL OF   1

FUI NO.             FREE FUI START
1                   .050000
2                   .060000
3                   .070000             DM NO.        4 PARAMETERS OF A DM        - NCM=    5
4                   .080000                1          .300000         .040000          .005000         .000000
5                   .090000                2          .050000         .020000          .005000         .000000
6                   .005000                3          .200000         .025000          .007000         .000000
7                   .006000                4          .100000         .018000          .009000         .000000
8                   .007000                5          .090000         .035000          .006000         .000000
9                   .008000
10                  .009000
```

Figure 3-4 GASP IV Input Echo Check and User Input Data Incorporated in MUXDA.

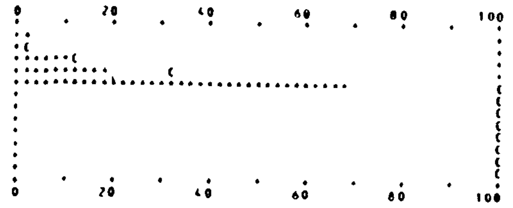
```
                                         **GASP FILE STORAGE AREA DUMP AT TIME      .0000    **

                                         MAXIMUM NUMBER OF ENTRIES IN FILE STORAGE AREA = 15

                                                  PRINTOUT OF FILE NUMBER    1
                                                     TNOW =      .0000
                                                     QQTIM=      .0000

                                                      FILE CONTENTS
ENTRY   1    =        .0000             .1010+03        .0000
ENTRY   2    =        .3193-01          .4020+03        .0000
ENTRY   3    =        .8715-01          .4040+03        .0000
ENTRY   4    =        .1000+00          .1020+03        .0000
ENTRY   5    =        .1140+00          .4050+03        .0000
ENTRY   6    =        .2000+00          .1030+03        .0000
ENTRY   7    =        .2036+00          .4030+03        .0000
ENTRY   8    =        .3000+00          .1040+03        .0000
ENTRY   9    =        .3373+00          .4010+03        .0000
ENTRY  10    =        .4000+00          .1050+03        .0000
ENTRY  11    =        .5000+00          .1060+03        .0000
ENTRY  12    =        .6000+00          .1070+03        .0000
ENTRY  13    =        .7000+00          .1080+03        .0000
ENTRY  14    =        .8000+00          .1090+03        .0000
ENTRY  15    =        .9000+00          .1100+03        .0000

                                                  PRINTOUT OF FILE NUMBER    2
                                                     TNOW =      .0000
                                                     QQTIM=      .0000

                                                     THE FILE IS EMPTY
```

Figure 3-5 File Printout at Time 0 for MUXDA in First Run.

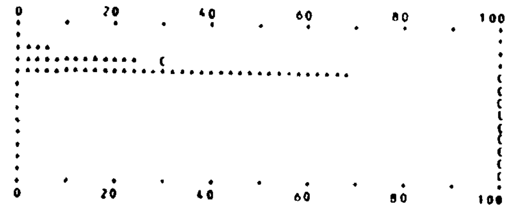
```
                                **GASP FILE STORAGE AREA DUMP AT TIME     .1000+03**

                                MAXIMUM NUMBER OF ENTRIES IN FILE STORAGE AREA = 28

                                             PRINTOUT OF FILE NUMBER    1
                                                 TNOW =     .1000+03
                                                 QQTIM=     .1000+03

                                       TIME PERIOD FOR STATISTICS    .1000+03
                                       AVERAGE NUMBER IN FILE        15.6913
                                       STANDARD DEVIATION               .4637
                                       MAXIMUM NUMBER IN FILE        16

                                                   FILE CONTENTS
ENTRY    1     =       .1000+03         .4020+03         .0000
ENTRY    2     =       .1000+03         .4010+03         .0000
ENTRY    3     =       .1000+03         .4040+03         .0000
ENTRY    4     =       .1000+03         .2010+03         .5000-01
ENTRY    5     =       .1001+03         .1020+03         .0000
ENTRY    6     =       .1001+03         .4050+03         .0000
ENTRY    7     =       .1001+03         .4030+03         .0000
ENTRY    8     =       .1002+03         .1030+03         .0000
ENTRY    9     =       .1003+03         .1040+03         .0000
ENTRY   10     =       .1004+03         .1050+03         .0000
ENTRY   11     =       .1005+03         .1060+03         .0000
ENTRY   12     =       .1006+03         .1070+03         .0000
ENTRY   13     =       .1007+03         .1080+03         .0000
ENTRY   14     =       .1008+03         .1090+03         .0000
ENTRY   15     =       .1009+03         .1100+03         .0000
ENTRY   16     =       .1010+03         .1010+03         .0000

                                             PRINTOUT OF FILE NUMBER    2
                                                 TNOW =     .1000+03
                                                 QQTIM=     .9999+02

                                       TIME PERIOD FOR STATISTICS    .1000+03
                                       AVERAGE NUMBER IN FILE          2.9019
                                       STANDARD DEVIATION              2.1125
                                       MAXIMUM NUMBER IN FILE        12

                                                   FILE CONTENTS
ENTRY    1     =       .9000-02         .4000+01         .9995+02
ENTRY    2     =       .6000-02         .5000+01         .9999+02
```

Figure 3-5 (Continued).

```
                         **HISTOGRAM NUMBER  1**
                               FUI 1S

OBSV    RELA    CUML       UPPER
FREQ    FREQ    FREQ     CELL LIMIT

   2    .020    .020     .1000-01
   0    .000    .020     .2000-01
  11    .109    .129     .3000-01
  19    .188    .317     .4000-01
  69    .683   1.000     .5000-01
   0    .000   1.000     .6000-01
   0    .000   1.000     .7000-01
   0    .000   1.000     .8000-01
   0    .000   1.000     .9000-01
   0    .000   1.000     .1000+00
   0    .000   1.000     .1100+00
   0    .000   1.000      INF
 ---
 101
```



```
                         **HISTOGRAM NUMBER  2**
                               FUI 2S

OBSV    RELA    CUML       UPPER
FREQ    FREQ    FREQ     CELL LIMIT

   1    .010    .010     .1000-01
   6    .060    .070     .2000-01
  24    .240    .310     .3000-01
  69    .690   1.000     .4000-01
   0    .000   1.000     .5000-01
   0    .000   1.000     .6000-01
   0    .000   1.000     .7000-01
   0    .000   1.000     .8000-01
   0    .000   1.000     .9000-01
   0    .000   1.000     .1000+00
   0    .000   1.000     .1100+00
   0    .000   1.000      INF
 ---
 100
```



```
                         **HISTOGRAM NUMBER  3**
                               FUI 3S

OBSV    RELA    CUML       UPPER
FREQ    FREQ    FREQ     CELL LIMIT

   3    .030    .030     .1000-01
   2    .020    .050     .2000-01
  95    .950   1.000     .3000-01
   0    .000   1.000     .4000-01
   0    .000   1.000     .5000-01
   0    .000   1.000     .6000-01
   0    .000   1.000     .7000-01
   0    .000   1.000     .8000-01
   0    .000   1.000     .9000-01
   0    .000   1.000     .1000+00
   0    .000   1.000     .1100+00
   0    .000   1.000      INF
 ---
 100
```

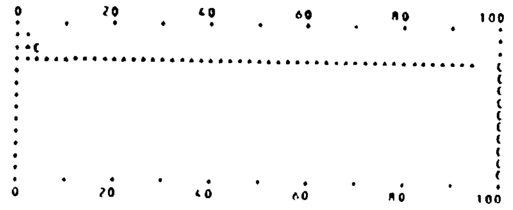
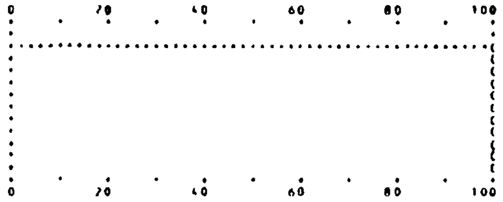
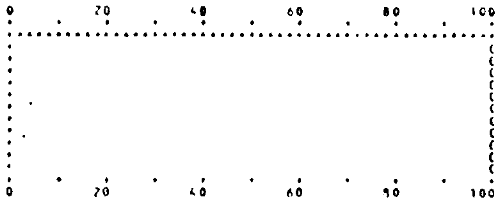


Figure 3-5 (Continued).

```
                                   **HISTOGRAM NUMBER  4**
                                          FUI 4S

 OBSV     RELA     CUML        UPPER
 FREQ     FREQ     FREQ     CELL LIMIT

    1     .010     .010      .1000-01
   99     .990    1.000      .2000-01
    0     .000    1.000      .3000-01
    0     .000    1.000      .4000-01
    0     .000    1.000      .5000-01
    0     .000    1.000      .6000-01
    0     .000    1.000      .7000-01
    0     .000    1.000      .8000-01
    0     .000    1.000      .9000-01
    0     .000    1.000      .1000+00
    0     .000    1.000      .1100+00
    0     .000    1.000        INF
  ---
  100
```



```
                                   **HISTOGRAM NUMBER  5**
                                          FUI 5S

 OBSV     RELA     CUML        UPPER
 FREQ     FREQ     FREQ     CELL LIMIT

  100    1.000    1.000      .1000-01
    0     .000    1.000      .2000-01
    0     .000    1.000      .3000-01
    0     .000    1.000      .4000-01
    0     .000    1.000      .5000-01
    0     .000    1.000      .6000-01
    0     .000    1.000      .7000-01
    0     .000    1.000      .8000-01
    0     .000    1.000      .9000-01
    0     .000    1.000      .1000+00
    0     .000    1.000      .1100+00
    0     .000    1.000        INF
  ---
  100
```



```
                                   **HISTOGRAM NUMBER  6**
                                          FUI 6S

 OBSV     RELA     CUML        UPPER
 FREQ     FREQ     FREQ     CELL LIMIT

    1     .010     .010      .1000-01
    0     .000     .010      .2000-01
    0     .000     .010      .3000-01
    0     .000     .010      .4000-01
    6     .060     .070      .5000-01
   20     .200     .270      .6000-01
   32     .320     .590      .7000-01
   25     .250     .840      .8000-01
   13     .130     .970      .9000-01
    3     .030    1.000      .1000+00
    0     .000    1.000      .1100+00
    0     .000    1.000        INF
  ---
  100
```

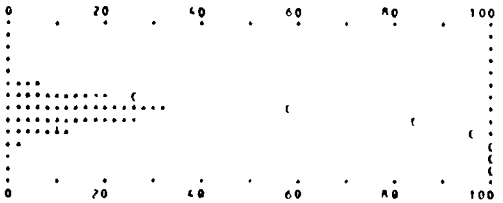
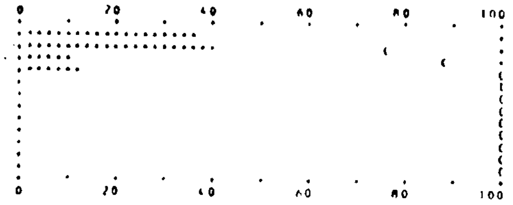
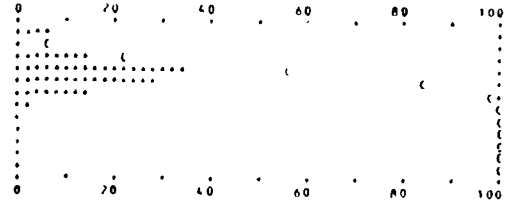


Figure 3-5 (Continued).

```
                              **HISTOGRAM NUMBER  7**
                                      FUI 7S

OBSV      RELA      CUML        UPPER
FREQ      FREQ      FREQ     CELL LIMIT
 36       .360      .360     .1000 01
 41       .410      .770     .2000 01
 11       .110      .880     .3000 01
 12       .120     1.000     .4000 01
  0       .000     1.000     .5000 01
  0       .000     1.000     .6000 01
  0       .000     1.000     .7000 01
  0       .000     1.000     .8000 01
  0       .000     1.000     .9000 01
  0       .000     1.000     .1000+00
  0       .000     1.000     .1100+00
  0       .000     1.000       INF
---
100
```



```
                              **HISTOGRAM NUMBER  8**
                                      FUI 8S

OBSV      RELA      CUML        UPPER
FREQ      FREQ      FREQ     CELL LIMIT
  6       .060      .060     .1000-01
  1       .010      .070     .2000-01
 15       .150      .220     .3000-01
 34       .340      .560     .4000-01
 28       .280      .840     .5000-01
 14       .140      .980     .6000-01
  2       .020     1.000     .7000-01
  0       .000     1.000     .8000-01
  0       .000     1.000     .9000-01
  0       .000     1.000     .1000+00
  0       .000     1.000     .1100+00
  0       .000     1.000       INF
---
100
```



```
                              **HISTOGRAM NUMBER  9**
                                      FUI 9S

OBSV      RELA      CUML        UPPER
FREQ      FREQ      FREQ     CELL LIMIT
  9       .090      .090     .1000-01
 29       .290      .380     .2000-01
 24       .240      .620     .3000-01
 21       .210      .830     .4000-01
 11       .110      .940     .5000-01
  4       .040      .980     .6000-01
  2       .020     1.000     .7000-01
  0       .000     1.000     .8000-01
  0       .000     1.000     .9000-01
  0       .000     1.000     .1000+00
  0       .000     1.000     .1100+00
  0       .000     1.000       INF
---
100
```

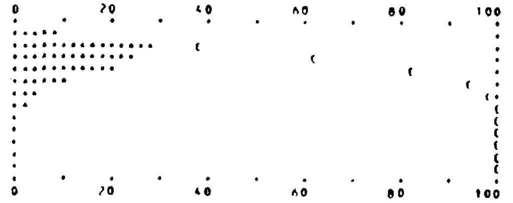
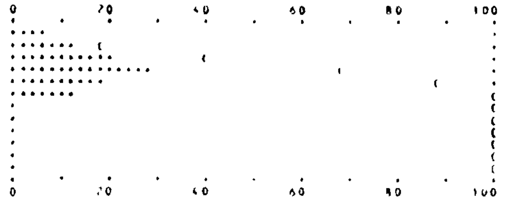


Figure 3-5 (Continued).

HISTOGRAM NUMBER 10

FUI 10S

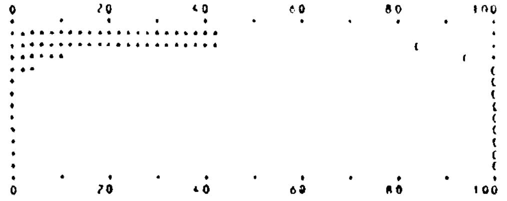
```
OBSV     RELA     CUML        UPPER
FREQ     FREQ     FREQ     CELL LIMIT

   6     .060     .060     .1000+01
  13     .130     .190     .2000+01
  21     .210     .400     .3000+01
  29     .290     .690     .4000+01
  19     .190     .880     .5000+01
  12     .120    1.000     .6000+01
   0     .000    1.000     .7000+01
   0     .000    1.000     .8000+01
   0     .000    1.000     .9000+01
   0     .000    1.000     .1000+00
   0     .000    1.000     .1100+00
   0     .000    1.000       INF
   -
 100
```



HISTOGRAM NUMBER 11

BR 1W

```
OBSV     RELA     CUML        UPPER
FREQ     FREQ     FREQ     CELL LIMIT

 141     .425      425     .5000 01
 140     .422     .846     .1000+00
  34     .102     .949     .1500+00
  15     .045     .994     .2000+00
   2     .006    1.000     .2500+00
   0     .000    1.000     .3000+00
   0     .000    1.000     .3500+00
   0     .000    1.000     .4000+00
   0     .000    1.000     .4500+00
   0     .000    1.000     .5000+00
   0     .000    1.000     .5500+00
   0     .000    1.000       INF
  --
 332
```



HISTOGRAM NUMBER 12

BR 2W

```
OBSV     RELA     CUML        UPPER
FREQ     FREQ     FREQ     CELL LIMIT

 861     .432     .432     .5000-01
 805     .404     .837     .1000+00
 244     .123     .959     .1500+00
  75     .038     .997     .2000+00
   6     .003    1.000     .2500+00
   0     .000    1.000     .3000+00
   0     .000    1.000     .3500+00
   0     .000    1.000     .4000+00
   0     .000    1.000     .4500+00
   0     .000    1.000     .5000+00
   0     .000    1.000     .5500+00
   0     .000    1.000       INF
 ---
1991
```

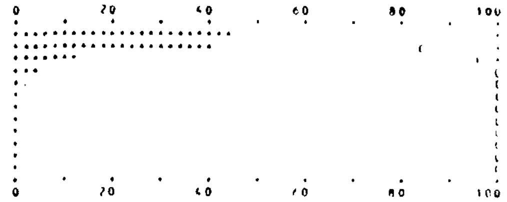
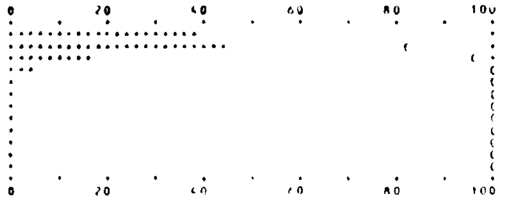
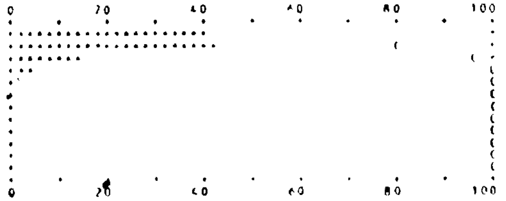


Figure 3-5 (Continued).

```
                         **HISTOGRAM NUMBER 13**
                                  BM 3W

 OBSV       RELA       CUML          UPPER
 FREQ       FREQ       FREQ       CELL LIMIT

  186       .374       .374       .5000-01
  219       .441       .815       .1000+00
   75       .151       .966       .1500+00
   15       .030       .996       .2000+00
    2       .004      1.000       .2500+00
    0       .000      1.000       .3000+00
    0       .000      1.000       .3500+00
    0       .000      1.000       .4000+00
    0       .000      1.000       .4500+00
    0       .000      1.000       .5000+00
    0       .000      1.000       .5500+00
    0       .000      1.000        INF
  ---
  497
```



```
                         **HISTOGRAM NUMBER 14**
                                  BM 4W

 OBSV       RELA       CUML          UPPER
 FREQ       FREQ       FREQ       CELL LIMIT

  393       .396       .396       .5000-01
  410       .413       .809       .1000+00
  141       .142       .951       .1500+00
   44       .044       .995       .2000+00
    5       .005      1.000       .2500+00
    0       .000      1.000       .3000+00
    0       .000      1.000       .3500+00
    0       .000      1.000       .4000+00
    0       .000      1.000       .4500+00
    0       .000      1.000       .5000+00
    0       .000      1.000       .5500+00
    0       .000      1.000        INF
  ---
  993
```



```
                         **HISTOGRAM NUMBER 15**
                                  BM 5W

 OBSV       RELA       CUML          UPPER
 FREQ       FREQ       FREQ       CELL LIMIT

  428       .388       .388       .5000-01
  468       .423       .811       .1000+00
  158       .143       .956       .1500+00
   47       .043       .999       .2000+00
    1       .001      1.000       .2500+00
    0       .000      1.000       .3000+00
    0       .000      1.000       .3500+00
    0       .000      1.000       .4000+00
    0       .000      1.000       .4500+00
    0       .000      1.000       .5000+00
    0       .000      1.000       .5500+00
    0       .000      1.000        INF
 ----
 1107
```

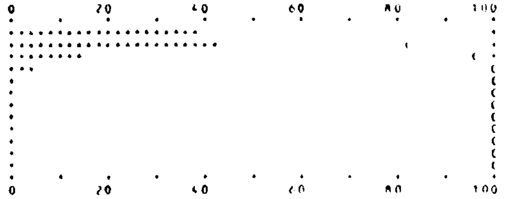


Figure 3-5 (Continued).

```
                                    **HISTOGRAM NUMBER 16**
                                           DM 6W

OBSV      RELA      CUML        UPPER
FREQ      FREQ      FREQ     CELL LIMIT      0   .   20   .   40   .   60   .   80   .   100
                                             .       .        .       .        .        .

NO VALUES RECORDED.

                                    **HISTOGRAM NUMBER 17**
                                           DM 7W

OBSV      RELA      CUML        UPPER
FREQ      FREQ      FREQ     CELL LIMIT      0   .   20   .   40   .   60   .   80   .   100
                                             .       .        .       .        .        .

NO VALUES RECORDED.

                                    **HISTOGRAM NUMBER 18**
                                           DM 8W

OBSV      RELA      CUML        UPPER
FREQ      FREQ      FREQ     CELL LIMIT      0   .   20   .   40   .   60   .   80   .   100
                                             .       .        .       .        .        .

NO VALUES RECORDED.

                                    **HISTOGRAM NUMBER 19**
                                           DM 9W

OBSV      RELA      CUML        UPPER
FREQ      FREQ      FREQ     CELL LIMIT      0   .   20   .   40   .   60   .   80   .   100
                                             .       .        .       .        .        .

NO VALUES RECORDED.

                                    **HISTOGRAM NUMBER 20**
                                           DM 10W

OBSV      RELA      CUML        UPPER
FREQ      FREQ      FREQ     CELL LIMIT      0   .   20   .   40   .   60   .   80   .   100
                                             .       .        .       .        .        .
```

Figure 3-5 (Continued).

```
**GASP FILE STORAGE AREA DUMP AT TIME     .1000+03**

MAXIMUM NUMBER OF ENTRIES IN FILE STORAGE AREA = 28

                                          PRINTOUT OF FILE NUMBER    1
                                               TNOW =      .1000+03
                                               QQTIM=      .1000+03

                                    TIME PERIOD FOR STATISTICS     .1000+03
                                    AVERAGE NUMBER IN FILE         15.6931
                                    STANDARD DEVIATION               .4629
                                    MAXIMUM NUMBER IN FILE         16

                                             FILE CONTENTS
ENTRY   1    =      .1000+03        .4050+03        .5000-01
ENTRY   2    =      .1000+03        .4020+03        .5000-01
ENTRY   3    =      .1000+03        .2010+03        .5000-01
ENTRY   4    =      .1001+03        .4040+03        .5000-01
ENTRY   5    =      .1001+03        .1020+03        .5000-01
ENTRY   6    =      .1001+03        .4010+03        .5000-01
ENTRY   7    =      .1002+03        .4030+03        .5000-01
ENTRY   8    =      .1002+03        .1030+03        .5000-01
ENTRY   9    =      .1003+03        .1040+03        .5000-01
ENTRY  10    =      .1004+03        .1050+03        .5000-01
ENTRY  11    =      .1005+03        .1060+03        .5000-01
ENTRY  12    =      .1006+03        .1070+03        .5000-01
ENTRY  13    =      .1007+03        .1080+03        .5000-01
ENTRY  14    =      .1008+03        .1090+03        .5000-01
ENTRY  15    =      .1009+03        .1100+03        .5000-01
ENTRY  16    =      .1010+03        .1010+03        .5000-01

                                          PRINTOUT OF FILE NUMBER    2
                                               TNOW =      .1000+03
                                               QQTIM=      .9998+02

                                    TIME PERIOD FOR STATISTICS     .1000+03
                                    AVERAGE NUMBER IN FILE          2.9605
                                    STANDARD DEVIATION              2.1782
                                    MAXIMUM NUMBER IN FILE         12

                                             FILE CONTENTS
ENTRY   1    =      .5000-02        .2000+01        .9994+02
ENTRY   2    =      .7000-02        .3000+01        .9994+02
ENTRY   3    =      .9000-02        .4000+01        .9996+02
ENTRY   4    =      .5000-02        .2000+01        .9998+02
```

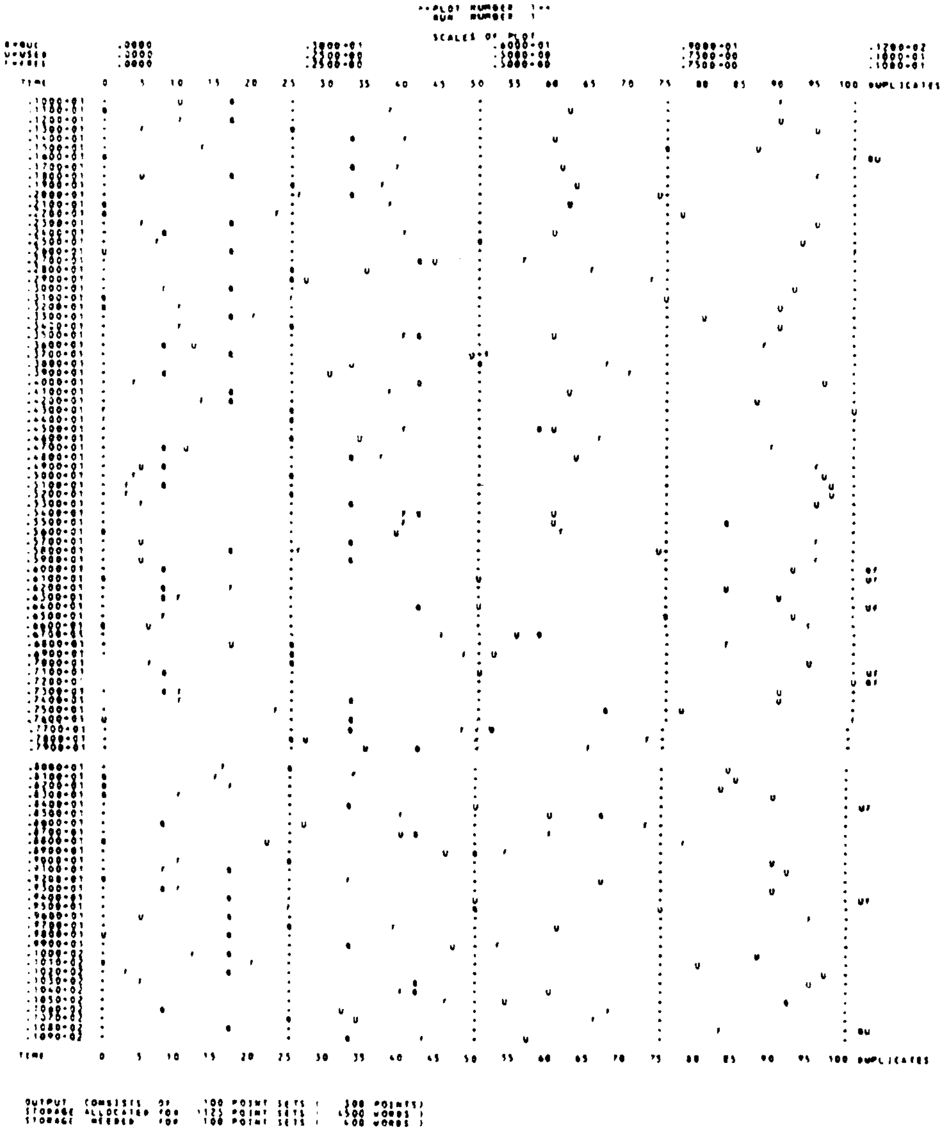
Figure 3-5 (Continued).

```
                                                                        **TABLE NUMBER    1**
                                                                          RUN   NUMBER    1
TIME          QUE           USED          FREE
 .1000+01      .2000+01      .1000+00      .9000+00
 .1100+01      .0000         .6250+00      .3750+00
 .1200+01      .2000+01      .9000+00      .1000+00
 .1300+01      .3000+01      .9500+00      .5000-01
 .1400+01      .4000+01      .6000+00      .4000+00
 .1500+01      .9000+01      .8737+00      .1263+00
 .1600+01      .0000         .0000         .1000+01
 .1700+01      .4000+01      .6129+00      .3871+00
 .1800+01      .2000+01      .5435-01      .9457+00
 .1900+01      .3000+01      .6264+00      .3736+00
 .2000+01      .4000+01      .7400+00      .2600+00
 .2100+01      .0000         .6250+00      .3750+00
 .2200+01      .0000         .7667+00      .2333+00
 .2300+01      .2000+01      .9500+00      .5000-01
 .2400+01      .1000+01      .6000+00      .4000+00
 .2500+01      .6000+01      .9263+00      .7368-01
 .2600+01      .2000+01      .0000         .1000+01
 .2700+01      .5000+01      .4409+00      .5591+00
 .2800+01      .3000+01      .3478+00      .6522+00
 .2900+01      .3000+01      .2747+00      .7253+00
 .3000+01      .2000+01      .9200+00      .8000-01
 .3100+01      .0000         .7500+00      .2500+00
 .3200+01      .0000         .9000+00      .1000+00
 .3300+01      .2000+01      .8000+00      .2000+00
 .3400+01      .3000+01      .9000+00      .1000+00
 .3500+01      .5000+01      .6000+00      .4000+00
 .3600+01      .1000+01      .1170+00      .8830+00
 .3700+01      .2000+01      .4946+00      .5054+00
 .3800+01      .6000+01      .3261+00      .6739+00
 .3900+01      .1000+01      .2967+00      .7033+00
 .4000+01      .5000+01      .9600+00      .4000-01
 .4100+01      .2000+01      .6250+00      .3750+00
 .4200+01      .2000+01      .8667+00      .1333+00
 .4300+01      .3000+01      .1000+01      .2980-05
 .4400+01      .3000+01      .1000+01      .2980-05
 .4500+01      .7000+01      .6000+00      .4000+00
 .4600+01      .3000+01      .3404+00      .6596+00
 .4700+01      .1000+01      .1075+00      .8925+00
 .4800+01      .4000+01      .6304+00      .3696+00
 .4900+01      .1000+01      .5495-01      .9451+00
 .5000+01      .3000+01      .9600+00      .4000-01
 .5100+01      .1000+01      .9750+00      .2500-01
 .5200+01      .3000+01      .9667+00      .3333-01
 .5300+01      .4000+01      .9500+00      .5000-01
 .5400+01      .5000+01      .6000+00      .4000+00
 .5500+01      .1000+02      .6000+00      .4000+00
 .5600+01      .0000         .3936+00      .6064+00
 .5700+01      .4000+01      .5376-01      .9462+00
 .5800+01      .2000+01      .7391+00      .2609+00
 .5900+01      .4000+01      .5495-01      .9451+00
 .6000+01      .1000+01      .9200+00      .8000-01
 .6100+01      .0000         .5000+00      .5000+00
 .6200+01      .1000+01      .8333+00      .1667+00
 .6300+01      .1000+01      .9000+00      .1000+00
 .6400+01      .5000+01      .5000+00      .5000+00
 .6500+01      .9000+01      .9158+00      .8421-01
 .6600+01      .0000         .6383-01      .9362+00
 .6700+01      .7000+01      .5484+00      .4516+00
 .6800+01      .3000+01      .1739+00      .8261+00
 .6900+01      .3000+01      .5165+00      .4835+00
 .7000+01      .3000+01      .9400+00      .6000-01
 .7100+01      .1000+01      .5000+00      .5000+00
 .7200+01      .0000         .1000+01      .1967-05
 .7300+01      .1000+01      .9000+00      .1000+00
 .7400+01      .4000+01      .9000+00      .1000+00
 .7500+01      .8000+01      .7684+00      .2316+00
 .7600+01      .4000+01      .0000         .1000+01
 .7700+01      .4000+01      .5161+00      .4839+00
 .7800+01      .3000+01      .2717+00      .7283+00
 .7900+01      .5000+01      .3516+00      .6484+00
 .8000+01      .3000+01      .8400+00      .1600+00
 .8100+01      .0000         .8500+00      .1500+00
 .8200+01      .0000         .8333+00      .1667+00
 .8300+01      .0000         .9000+00      .1000+00
 .8400+01      .4000+01      .5000+00      .5000+00
 .8500+01      .8000+01      .6000+00      .4000+00
```

Figure 3-5 (Continued).

```
.8600+01     .1000+01     .2660+00     .7340+00
.8700+01     .5000+01     .3978+00     .6022+00
.8800+01     .0000        .2174+00     .7826+00
.8900+01     .6000+01     .4615+00     .5385+00
.9000+01     .3000+01     .9000+00     .1000+00
.9100+01     .2000+01     .9250+00     .7500-01
.9200+01     .0000        .6667+00     .3333+00
.9300+01     .1000+01     .9000+00     .1000+00
.9400+01     .2000+01     .5000+00     .5000+00
.9500+01     .6000+01     .7474+00     .2526+00
.9600+01     .2000+01     .5319-01     .9468+00
.9700+01     .3000+01     .6129+00     .3871+00
.9800+01     .2000+01     .0000        .1000+01
.9900+01     .4000+01     .4725+00     .5275+00
.1000+02     .2000+01     .8800+00     .1200+00
.1010+02     .0000        .8000+00     .2000+00
.1020+02     .2000+01     .9667+00     .3334-01
.1030+02     .5000+01     .9500+00     .5000-01
.1040+02     .5000+01     .6000+00     .4000+00
.1050+02     .1100+02     .5368+00     .4632+00
.1060+02     .1000+01     .3191+00     .6809+00
.1070+02     .3000+01     .3441+00     .6559+00
.1080+02     .2000+01     .1739+00     .8261+00
.1090+02     .4000+01     .5714+00     .4286+00

  MINIMUM    .0000        .0000        .1987-05
  MAXIMUM    .1100+02     .1000+01     .1000+01
```

Figure 3-5 (Continued).



```
                              **PLOT NUMBER  1**
                                RUN  NUMBER  1

                                 SCALES OF PLOT
Q=QUE       .0000            .3000+01            .6000+01            .9000+01            .1200+02
U=USED      .0000            .2500+00            .5000+00            .7500+00            .1000+01
F=FULL      .0000            .2500+00            .5000+00            .7500+00            .1000+01

  TIME       0   5  10  15  20  25  30  35  40  45  50  55  60  65  70  75  80  85  90  95 100 DUPLICATES

  TIME       0   5  10  15  20  25  30  35  40  45  50  55  60  65  70  75  80  85  90  95 100 DUPLICATES

OUTPUT   CONSISTS   OF      100 POINT SETS (   300 POINTS)
STORAGE  ALLOCATED  FOR    1125 POINT SETS (  4500 WORDS )
STORAGE   NEEDED    FOR     100 POINT SETS (   400 WORDS )
```

Figure 3-6 Plot Output for MUXDA.

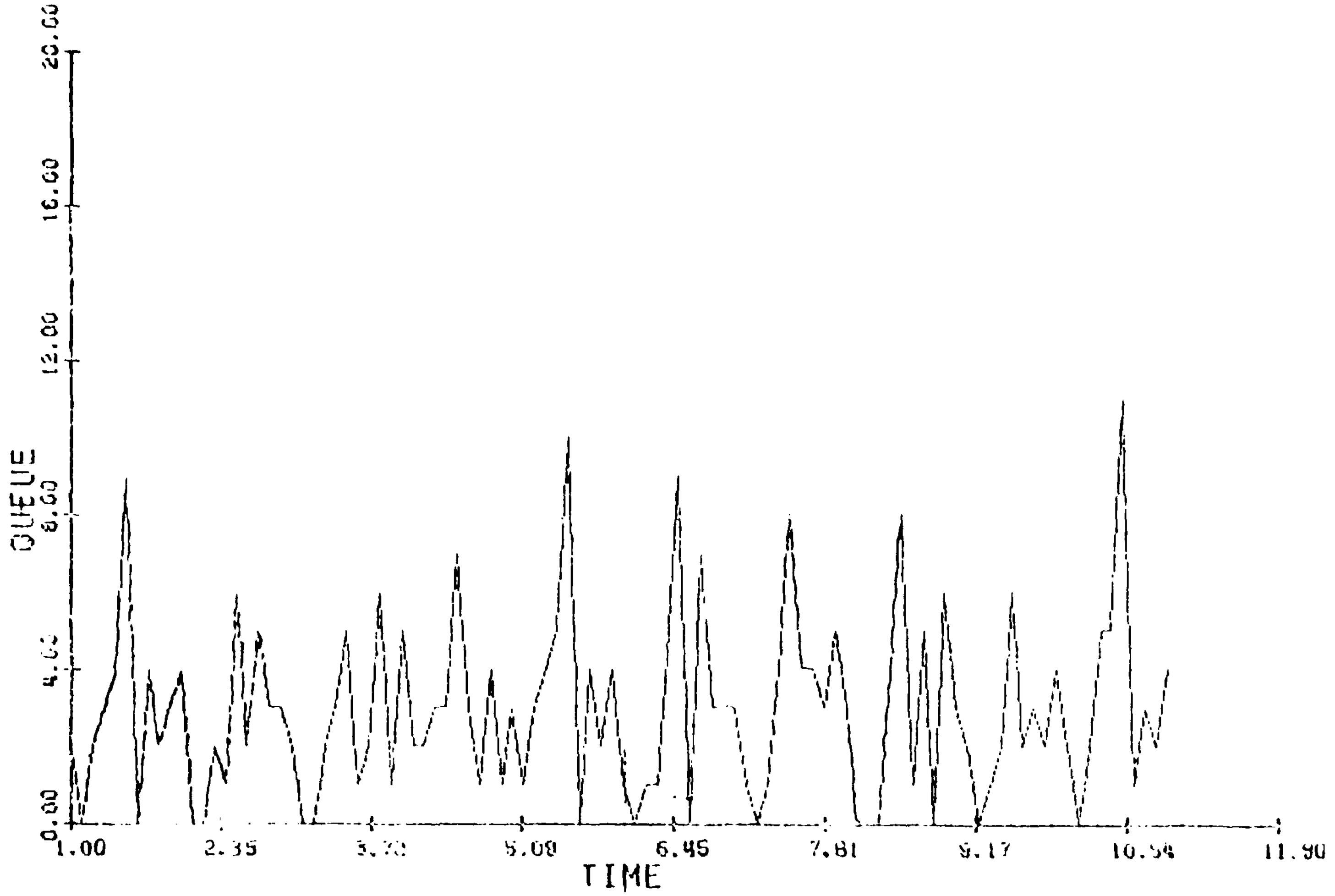


Figure 3-7 Variation of Queue Length with Time in MUXDA (First Run).

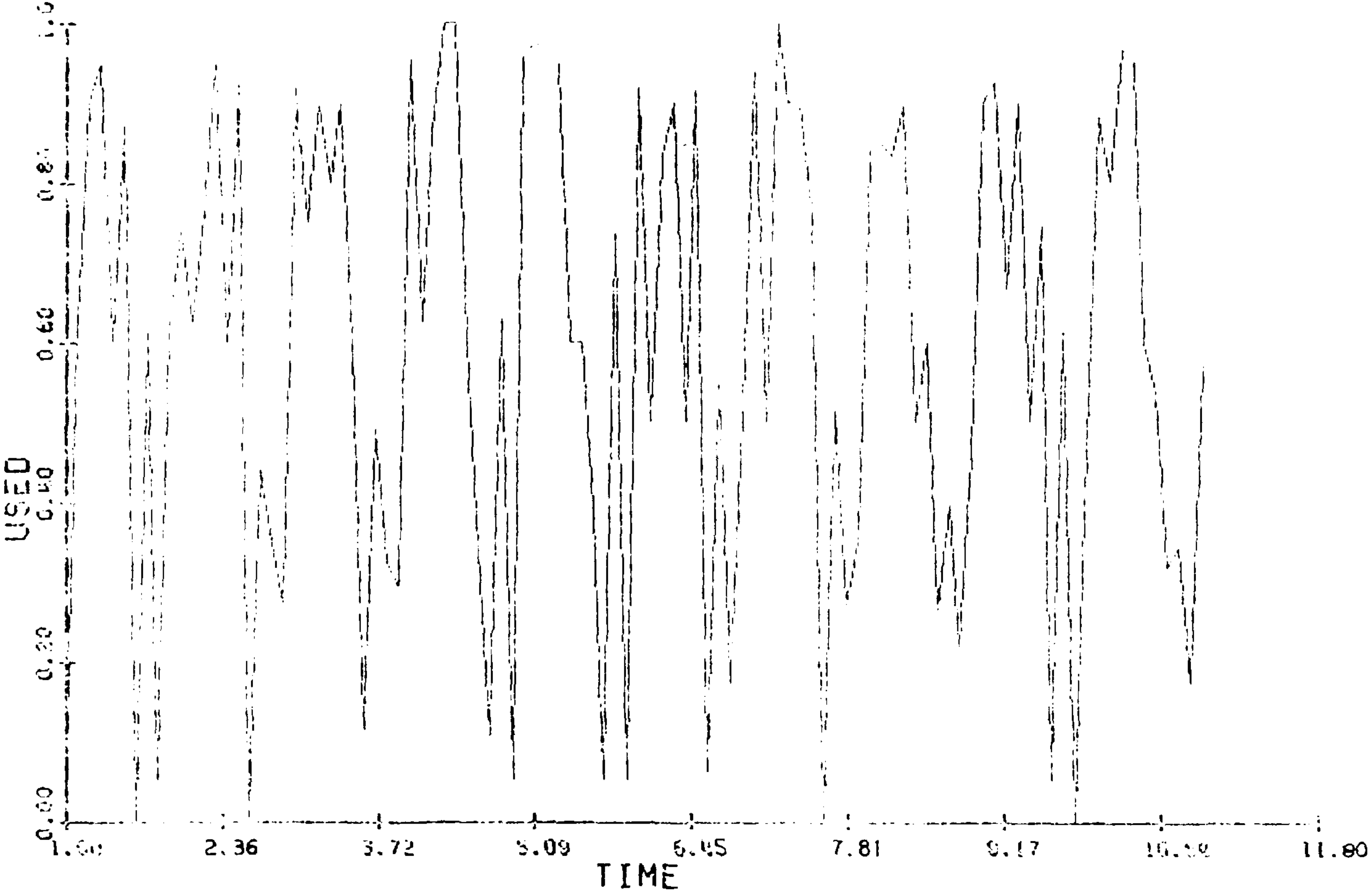


Figure 3-8 Variation of the Percentage of Interval Time Available for Demand Message Transmission with Time in MUXDA (First Run).

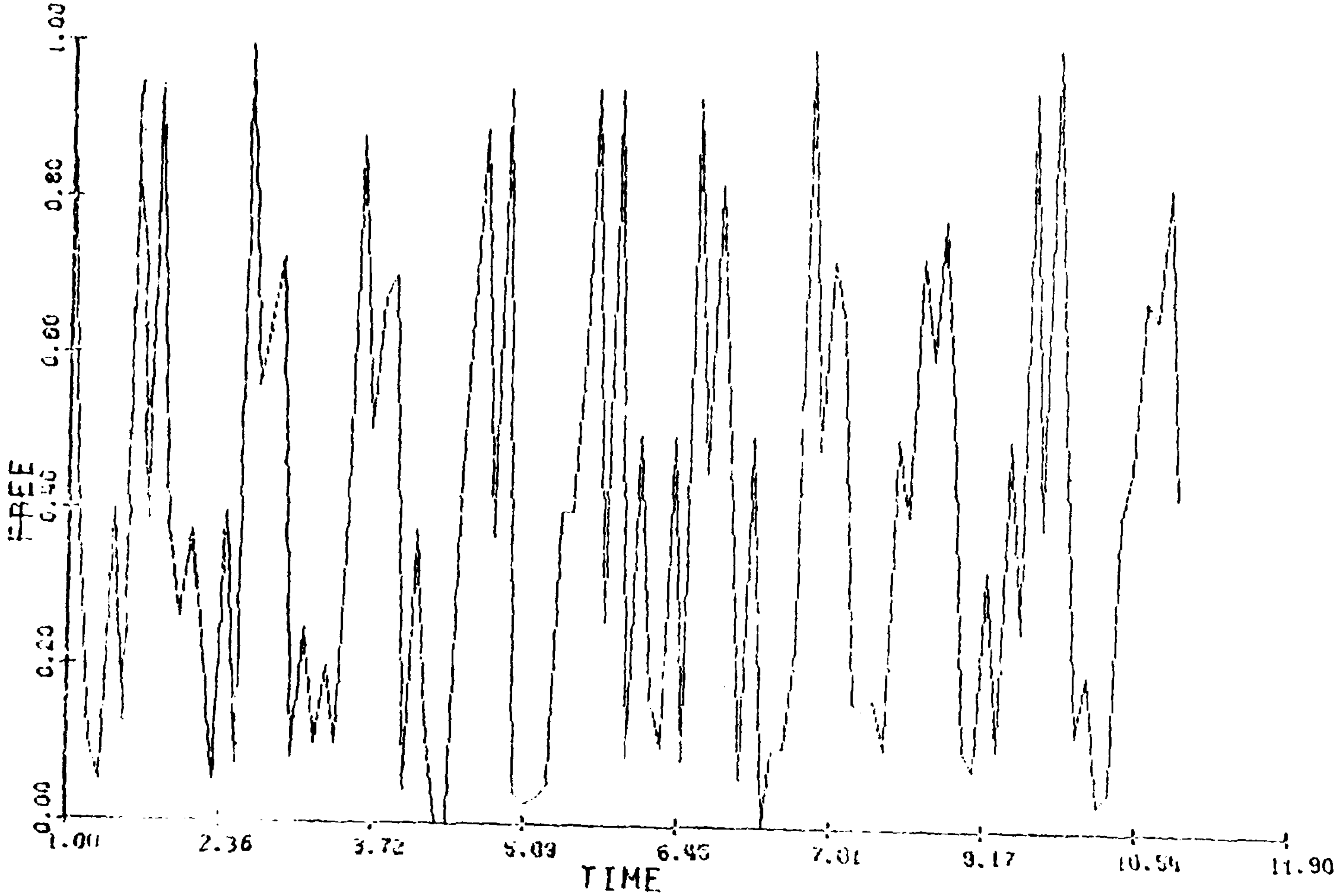
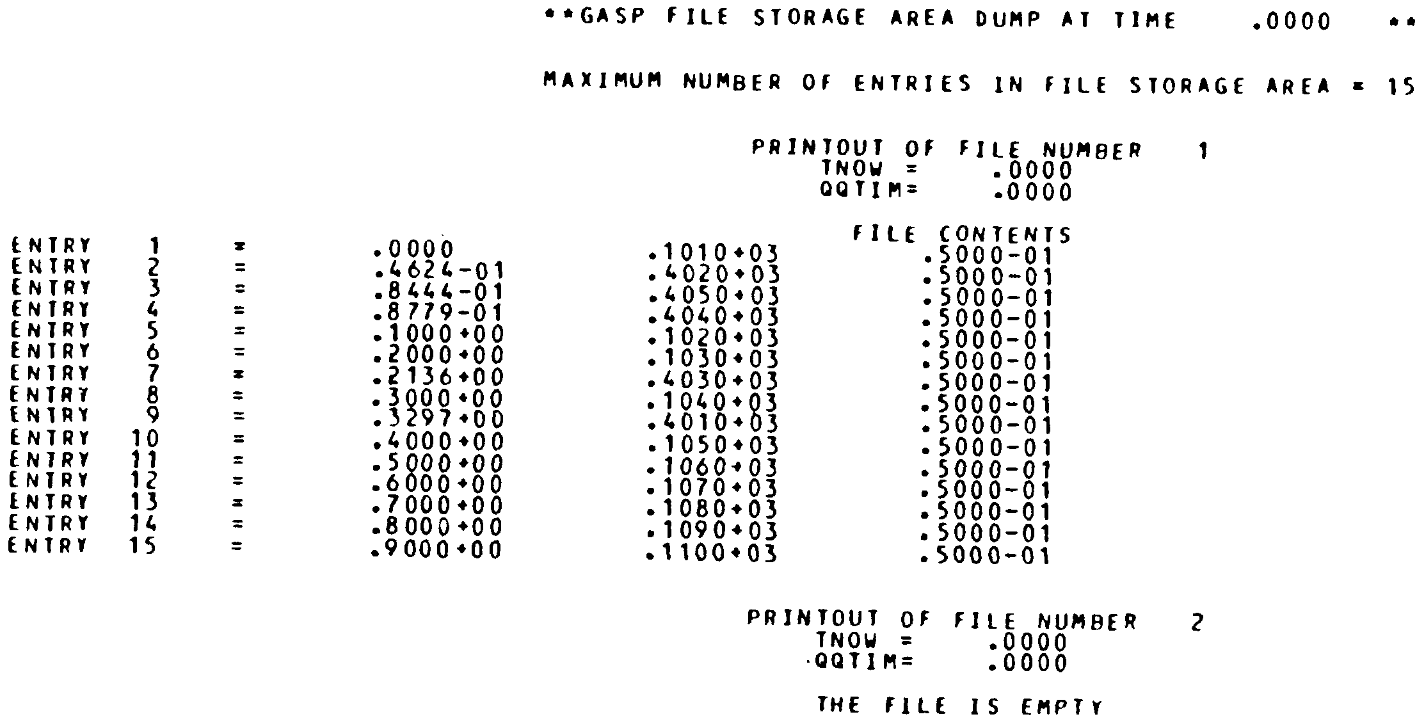


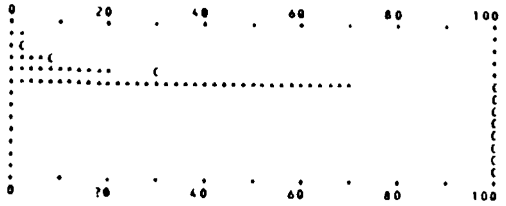
Figure 3-9 Variation of Bus Idle Time with Time in MUXDA (First Run).



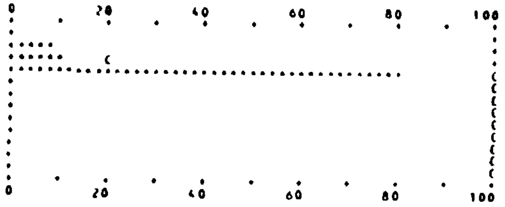
```
                           **GASP FILE STORAGE AREA DUMP AT TIME     .0000    **

                           MAXIMUM NUMBER OF ENTRIES IN FILE STORAGE AREA = 15

                                            PRINTOUT OF FILE NUMBER    1
                                              TNOW =     .0000
                                              QQTIM=     .0000

                                                FILE CONTENTS
ENTRY   1   =       .0000        .1010+03        .5000-01
ENTRY   2   =       .4624-01     .4020+03        .5000-01
ENTRY   3   =       .8444-01     .4050+03        .5000-01
ENTRY   4   =       .8779-01     .4040+03        .5000-01
ENTRY   5   =       .1000+00     .1020+03        .5000-01
ENTRY   6   =       .2000+00     .1030+03        .5000-01
ENTRY   7   =       .2136+00     .4030+03        .5000-01
ENTRY   8   =       .3000+00     .1040+03        .5000-01
ENTRY   9   =       .3297+00     .4010+03        .5000-01
ENTRY  10   =       .4000+00     .1050+03        .5000-01
ENTRY  11   =       .5000+00     .1060+03        .5000-01
ENTRY  12   =       .6000+00     .1070+03        .5000-01
ENTRY  13   =       .7000+00     .1080+03        .5000-01
ENTRY  14   =       .8000+00     .1090+03        .5000-01
ENTRY  15   =       .9000+00     .1100+03        .5000-01

                                            PRINTOUT OF FILE NUMBER    2
                                              TNOW =     .0000
                                             .QQTIM=     .0000

                                              THE FILE IS EMPTY
```

Figure 3-10 File Printout at Time 0 for MUXDA in Second Pun.

```
**HISTOGRAM NUMBER 1**
        FUI 1S

 OBSV     RELA     CUML       UPPER
 FREQ     FREQ     FREQ     CELL LIMIT

    2     .020     .020     .1000-01
    0     .000     .020     .2000-01
    7     .069     .089     .3000-01
   21     .208     .297     .4000-01
   71     .703    1.000     .5000-01
    0     .000    1.000     .6000-01
    0     .000    1.000     .7000-01
    0     .000    1.000     .8000-01
    0     .000    1.000     .9000-01
    0     .000    1.000     .1000+00
    0     .000    1.000     .1100+00
    0     .000    1.000      INF
  ---
  101
```



```
**HISTOGRAM NUMBER 2**
        FUI 2S

 OBSV     RELA     CUML       UPPER
 FREQ     FREQ     FREQ     CELL LIMIT

    1     .010     .010     .1000-01
    8     .080     .090     .2000-01
   11     .110     .200     .3000-01
   80     .800    1.000     .4000-01
    0     .000    1.000     .5000-01
    0     .000    1.000     .6000-01
    0     .000    1.000     .7000-01
    0     .000    1.000     .8000-01
    0     .000    1.000     .9000-01
    0     .000    1.000     .1000+00
    0     .000    1.000     .1100+00
    0     .000    1.000      INF
  ---
  100
```



```
**HISTOGRAM NUMBER 3**
        FUI 3S

 OBSV     RELA     CUML       UPPER
 FREQ     FREQ     FREQ     CELL LIMIT

    1     .010     .010     .1000-01
    6     .060     .070     .2000-01
   93     .930    1.000     .3000-01
    0     .000    1.000     .4000-01
    0     .000    1.000     .5000-01
    0     .000    1.000     .6000-01
    0     .000    1.000     .7000-01
    0     .000    1.000     .8000-01
    0     .000    1.000     .9000-01
    0     .000    1.000     .1000+00
    0     .000    1.000     .1100+00
    0     .000    1.000      INF
  ---
  100
```

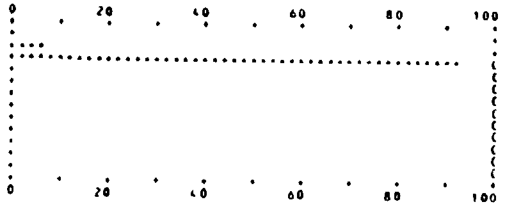
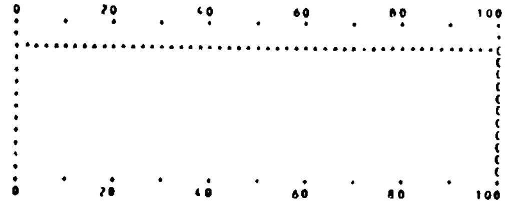
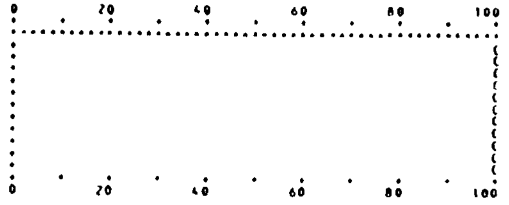


Figure 3-10 (Continued).

```
                         **HISTOGRAM NUMBER  4**
                                FUI 4S

OBSV    RELA    CUML       UPPER
FREQ    FREQ    FREQ    CELL LIMIT

   1    .010    .010     .1000-01
  99    .990   1.000     .2000-01
   0    .000   1.000     .3000-01
   0    .000   1.000     .4000-01
   0    .000   1.000     .5000-01
   0    .000   1.000     .6000-01
   0    .000   1.000     .7000-01
   0    .000   1.000     .8000-01
   0    .000   1.000     .9000-01
   0    .000   1.000     .1000+00
   0    .000   1.000     .1100+00
   0    .000   1.000      INF
 ---
 100
```



```
                         **HISTOGRAM NUMBER  5**
                                FUI 5S

OBSV    RELA    CUML       UPPER
FREQ    FREQ    FREQ    CELL LIMIT

 100   1.000   1.000     .1000-01
   0    .000   1.000     .2000-01
   0    .000   1.000     .3000-01
   0    .000   1.000     .4000-01
   0    .000   1.000     .5000-01
   0    .000   1.000     .6000-01
   0    .000   1.000     .7000-01
   0    .000   1.000     .8000-01
   0    .000   1.000     .9000-01
   0    .000   1.000     .1000+00
   0    .000   1.000     .1100+00
   0    .000   1.000      INF
 ---
 100
```



```
                         **HISTOGRAM NUMBER  6**
                                FUI 6S

OBSV    RELA    CUML       UPPER
FREQ    FREQ    FREQ    CELL LIMIT

   1    .010    .010     .1000-01
   0    .000    .010     .2000-01
   0    .000    .010     .3000-01
   4    .040    .050     .4000-01
  15    .150    .200     .5000-01
  22    .220    .420     .6000-01
  38    .380    .800     .7000-01
  14    .140    .940     .8000-01
   5    .050    .990     .9000-01
   1    .010   1.000     .1000+00
   0    .000   1.000     .1100+00
   0    .000   1.000      INF
 ---
 100
```

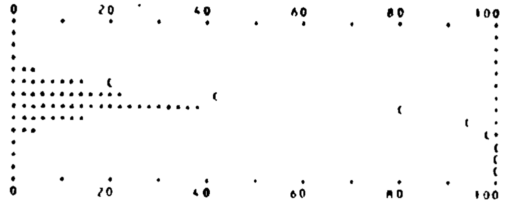
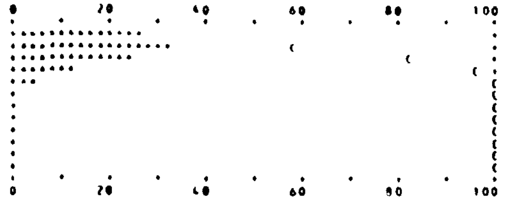
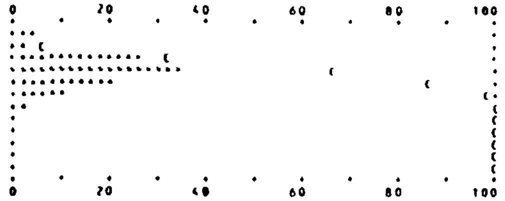


Figure 3-10 (Continued).

```
**HISTOGRAM NUMBER  7**
        FUI 7S

 OBSV     RELA     CUML       UPPER
 FREQ     FREQ     FREQ    CELL LIMIT

  26      .260     .260     .1000-01
  33      .330     .590     .2000-01
  24      .240     .830     .3000-01
  13      .130     .960     .4000-01
   4      .040    1.000     .5000-01
   0      .000    1.000     .6000-01
   0      .000    1.000     .7000-01
   0      .000    1.000     .8000-01
   0      .000    1.000     .9000-01
   0      .000    1.000     .1000+00
   0      .000    1.000     .1100+00
   0      .000    1.000      INF
 ---
 100
```



```
**HISTOGRAM NUMBER  8**
        FUI 8S

 OBSV     RELA     CUML       UPPER
 FREQ     FREQ     FREQ    CELL LIMIT

   4      .040     .040     .1000-01
   2      .020     .060     .2000-01
  26      .260     .320     .3000-01
  34      .340     .660     .4000-01
  21      .210     .870     .5000-01
  11      .110     .980     .6000-01
   2      .020    1.000     .7000-01
   0      .000    1.000     .8000-01
   0      .000    1.000     .9000-01
   0      .000    1.000     .1000+00
   0      .000    1.000     .1100+00
   0      .000    1.000      INF
 ---
 100
```



```
**HISTOGRAM NUMBER  9**
        FUI 9S

 OBSV     RELA     CUML       UPPER
 FREQ     FREQ     FREQ    CELL LIMIT

   8      .080     .080     .1000-01
  24      .240     .320     .2000-01
  29      .290     .610     .3000-01
  25      .250     .860     .4000-01
  10      .100     .960     .5000-01
   3      .030     .990     .6000-01
   1      .010    1.000     .7000-01
   0      .000    1.000     .8000-01
   0      .000    1.000     .9000-01
   0      .000    1.000     .1000+00
   0      .000    1.000     .1100+00
   0      .000    1.000      INF
 ---
 100
```

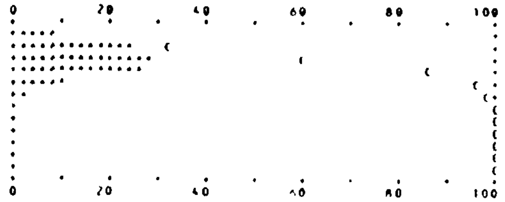
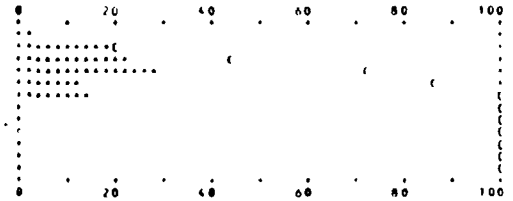
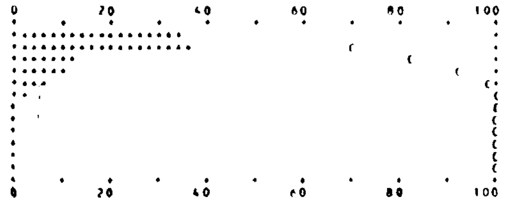


Figure 3-10 (Continued).

```
                    **HISTOGRAM NUMBER 10**
                          FUI lBS

 OBSV     RELA     CUML       UPPER
 FREQ     FREQ     FREQ     CELL LIMIT

    3     .030     .030     .1000-01
   18     .180     .210     .2000-01
   23     .230     .440     .3000-01
   29     .290     .730     .4000-01
   13     .130     .860     .5000-01
   14     .140    1.000     .6000-01
    0     .000    1.000     .7000-01
    0     .000    1.000     .8000-01
    0     .000    1.000     .9000-01
    0     .000    1.000     .1000+00
    0     .000    1.000     .1100+00
    0     .000    1.000      INF
  ---
  100
```



```
                    **HISTOGRAM NUMBER 11**
                          BM 1W

 OBSV     RELA     CUML       UPPER
 FREQ     FREQ     FREQ     CELL LIMIT

  114     .343     .343     .5000-01
  118     .355     .699     .1000+00
   41     .123     .822     .1500+00
   31     .093     .916     .2000+00
   20     .060     .976     .2500+00
    7     .021     .997     .3000+00
    1     .003    1.000     .3500+00
    0     .000    1.000     .4000+00
    0     .000    1.000     .4500+00
    0     .000    1.000     .5000+00
    0     .000    1.000     .5500+00
    0     .000    1.000      INF
  ---
  332
```



```
                    **HISTOGRAM NUMBER 12**
                          BM 2W

 OBSV     RELA     CUML       UPPER
 FREQ     FREQ     FREQ     CELL LIMIT

  676     .337     .337     .5000-01
  690     .344     .681     .1000+00
  329     .164     .845     .1500+00
  168     .084     .929     .2000+00
   97     .048     .977     .2500+00
   38     .019     .996     .3000+00
    6     .003     .999     .3500+00
    2     .001    1.000     .4000+00
    0     .000    1.000     .4500+00
    0     .000    1.000     .5000+00
    0     .000    1.000     .5500+00
    0     .000    1.000      INF
    -
 2006
```

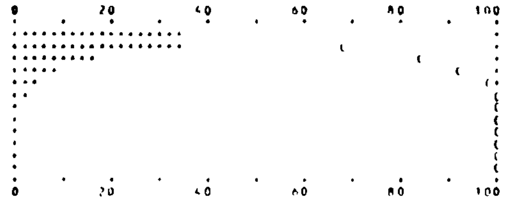
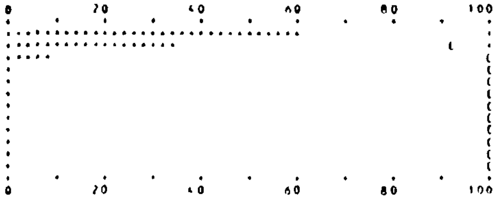
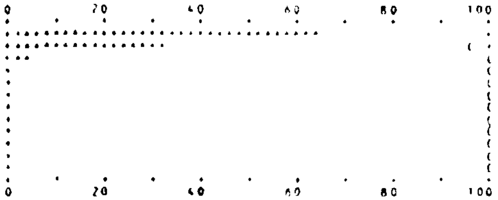


Figure 3-10 (Continued).

```
                              **HISTOGRAM NUMBER 13**
                                      DM 3W

OBSV      RELA      CUML        UPPER
FREQ      FREQ      FREQ     CELL LIMIT

 296      .594      .594     .5000 01
 165      .331      .926     .1000+00
  36      .072      .998     .1500+00
   1      .002     1.000     .2000+00
   0      .000     1.000     .2500+00
   0      .000     1.000     .3000+00
   0      .000     1.000     .3500+00
   0      .000     1.000     .4000+00
   0      .000     1.000     .4500+00
   0      .000     1.000     .5000+00
   0      .000     1.000     .5500+00
   0      .000     1.000       INF
 ---
 498
```



```
                              **HISTOGRAM NUMBER 14**
                                      DM 4W

OBSV      RELA      CUML        UPPER
FREQ      FREQ      FREQ     CELL LIMIT

 639      .641      .641     .5000 01
 326      .327      .968     .1000+00
  32      .032     1.000     .1500+00
   0      .000     1.000     .2000+00
   0      .000     1.000     .2500+00
   0      .000     1.000     .3000+00
   0      .000     1.000     .3500+00
   0      .000     1.000     .4000+00
   0      .000     1.000     .4500+00
   0      .000     1.000     .5000+00
   0      .000     1.000     .5500+00
   0      .000     1.000       INF
 ---
 997
```



```
                              **HISTOGRAM NUMBER 15**
                                      DM 5W

OBSV      RELA      CUML        UPPER
FREQ      FREQ      FREQ     CELL LIMIT

 614      .552      .552     .5000 01
 344      .309      .862     .1000+00
 108      .097      .959     .1500+00
  28      .025      .984     .2000+00
  18      .016     1.000     .2500+00
   0      .000     1.000     .3000+00
   0      .000     1.000     .3500+00
   0      .000     1.000     .4000+00
   0      .000     1.000     .4500+00
   0      .000     1.000     .5000+00
   0      .000     1.000     .5500+00
   0      .000     1.000       INF
   -
1112
```

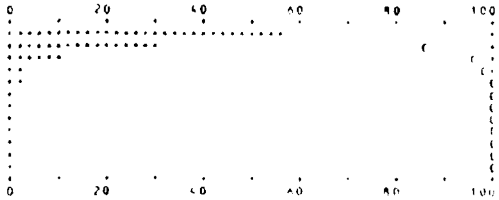


Figure 3-10 (Continued).

```
                              **HISTOGRAM NUMBER 16**
                                      DM 6W
OBSV      RELA      CUML        UPPER
FREQ      FREQ      FREQ      CELL LIMIT     0    .    20    .    40    .    60    .    80    .   100
                                             .         .          .         .         .          .

NO VALUES RECORDED.


                              **HISTOGRAM NUMBER 17**
                                      DM 7W
OBSV      RELA      CUML        UPPER
FREQ      FREQ      FREQ      CELL LIMIT     0    .    20         40    .    60    .    80    .   100
                                             .                    .         .         .          .

NO VALUES RECORDED.


                              **HISTOGRAM NUMBER 18**
                                      DM 8W
OBSV      RELA      CUML        UPPER
FREQ      FREQ      FREQ      CELL LIMIT     0    .    20    .    40    .    60    .    80    .   100
                                             .         .          .         .         .          .

NO VALUES RECORDED.


                              **HISTOGRAM NUMBER 19**
                                      DM 9W
OBSV      RELA      CUML        UPPER
FREQ      FREQ      FREQ      CELL LIMIT     0    .    20    .    40    .    60    .    80    .   100
                                             .         .          .         .         .          .

NO VALUES RECORDED.


                              **HISTOGRAM NUMBER 20**
                                      DM 10W
OBSV      RELA      CUML        UPPER
FREQ      FREQ      FREQ      CELL LIMIT     0    .    20    .    40    .    60    .    80    .   100
                                             .         .          .         .         .          .
```

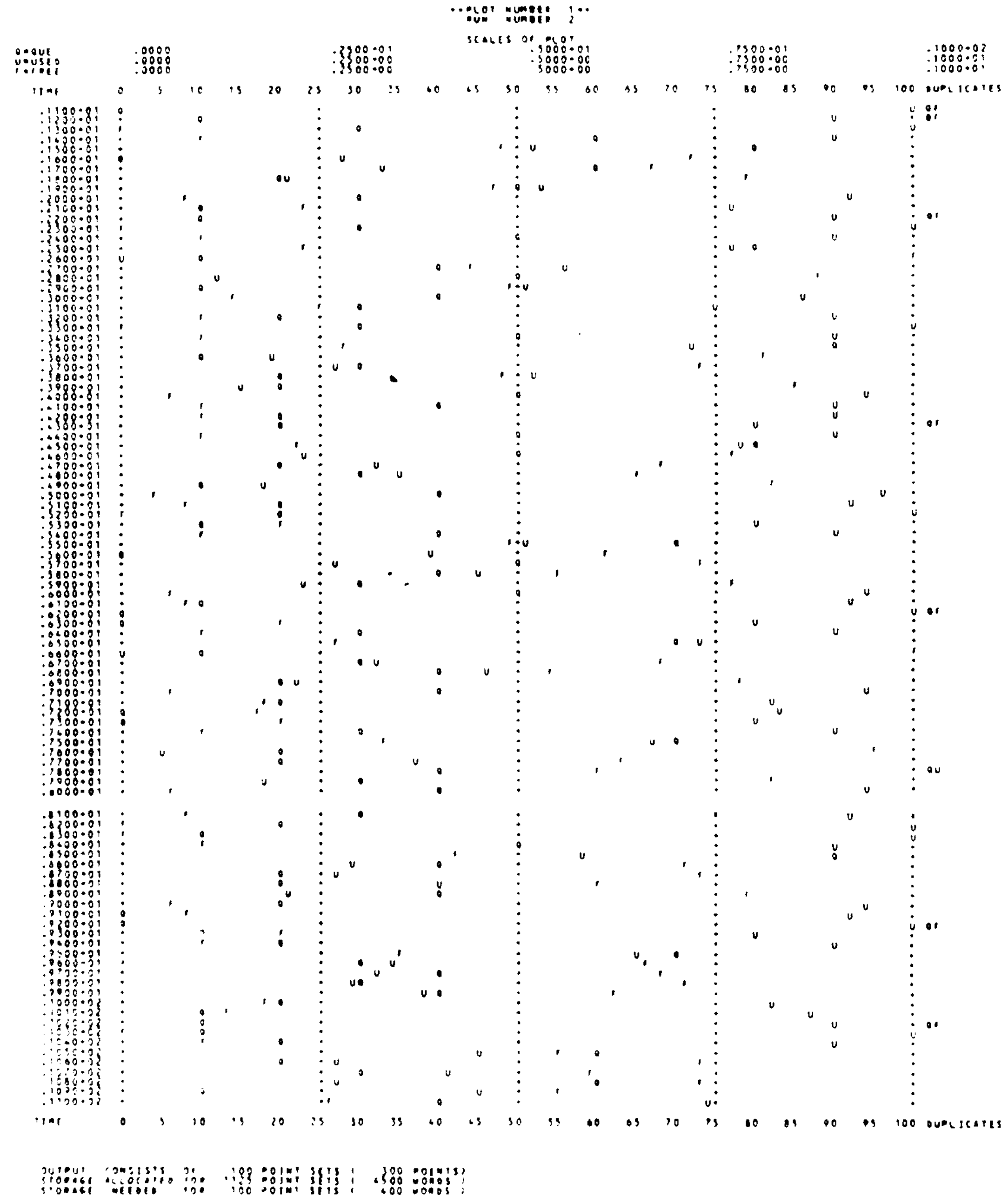
Figure 3-10 (Continued).

```
                                                  **TABLE NUMBER   1**
                                                    RUN  NUMBER    2

TIME             QUE              USED             FREE
  .1100+01        .0000            .1000+01         .1397-06
  .1200+01        .1000+01         .9000+00         .1000+00
  .1300+01        .3000+01         .1000+01         .1863-06
  .1400+01        .6000+01         .9000+00         .1000+00
  .1500+01        .8000+01         .5156+00         .4842+00
  .1600+01        .0000            .2766+00         .7234+00
  .1700+01        .6000+01         .3333+00         .6667+00
  .1800+01        .2000+01         .2065+00         .7935+00
  .1900+01        .5000+01         .5275+00         .4725+00
  .2000+01        .3000+01         .9200+00         .8000-01
  .2100+01        .1000+01         .7750+00         .2250+00
  .2200+01        .1000+01         .9000+00         .1000+00
  .2300+01        .3000+01         .1000+01         .7451-06
  .2400+01        .5000+01         .9000+00         .1000+00
  .2500+01        .8000+01         .7684+00         .2316+00
  .2600+01        .1000+01         .0000            .1000+01
  .2700+01        .4000+01         .5591+00         .4409+00
  .2800+01        .5000+01         .1196+00         .8804+00
  .2900+01        .1000+01         .5055+00         .4945+00
  .3000+01        .4000+01         .8600+00         .1400+00
  .3100+01        .3000+01         .7500+00         .2500+00
  .3200+01        .2000+01         .9000+00         .1000+00
  .3300+01        .3000+01         .1000+01         .2980-05
  .3400+01        .5000+01         .9000+00         .1000+00
  .3500+01        .9000+01         .7158+00         .2842+00
  .3600+01        .1000+01         .1915+00         .8085+00
  .3700+01        .3000+01         .2688+00         .7312+00
  .3800+01        .2000+01         .5217+00         .4783+00
  .3900+01        .2000+01         .1538+00         .6462+00
  .4000+01        .5000+01         .9400+00         .6000-01
  .4100+01        .4000+01         .9000+00         .1000+00
  .4200+01        .2000+01         .9000+00         .1000+00
  .4300+01        .2000+01         .8000+00         .2000+00
  .4400+01        .5000+01         .9000+00         .1000+00
  .4500+01        .8000+01         .7789+00         .2211+00
  .4600+01        .5000+01         .2340+00         .7660+00
  .4700+01        .2000+01         .3226+00         .6774+00
  .4800+01        .3000+01         .3478+00         .6522+00
  .4900+01        .1000+01         .1758+00         .8242+00
  .5000+01        .4000+01         .9600+00         .4000-01
  .5100+01        .2000+01         .9250+00         .7500-01
  .5200+01        .2000+01         .1000+01         .1987-05
  .5300+01        .1000+01         .8000+00         .2000+00
  .5400+01        .4000+01         .9000+00         .1000+00
  .5500+01        .7000+01         .5053+00         .4947+00
  .5600+01        .0000            .3936+00         .6064+00
  .5700+01        .5000+01         .2688+00         .7312+00
  .5800+01        .4000+01         .4457+00         .5543+00
  .5900+01        .3000+01         .2308+00         .7692+00
  .6000+01        .5000+01         .9400+00         .6000-01
  .6100+01        .1000+01         .9250+00         .7500-01
  .6200+01        .0000            .1000+01         .1987-05
  .6300+01        .0000            .8000+00         .2000+00
  .6400+01        .3000+01         .9000+00         .1000+00
  .6500+01        .7000+01         .7263+00         .2737+00
  .6600+01        .1000+01         .0000            .1000+01
  .6700+01        .3000+01         .3226+00         .6774+00
  .6800+01        .4000+01         .4565+00         .5435+00
  .6900+01        .2000+01         .2198+00         .7802+00
  .7000+01        .4000+01         .9400+00         .6000-01
  .7100+01        .2000+01         .8250+00         .1750+00
  .7200+01        .0000            .8333+00         .1667+00
  .7300+01        .0000            .8000+00         .2000+00
  .7400+01        .3000+01         .9000+00         .1000+00
  .7500+01        .7000+01         .6737+00         .3263+00
  .7600+01        .2000+01         .5319-01         .9468+00
  .7700+01        .2000+01         .3656+00         .6344+00
  .7800+01        .4000+01         .4022+00         .5978+00
  .7900+01        .3000+01         .1758+00         .8242+00
  .8000+01        .4000+01         .9400+00         .6000-01
  .8100+01        .3000+01         .9250+00         .7500-01
  .8200+01        .2000+01         .1000+01         .1987-05
  .8300+01        .1000+01         .1000+01         .2980-05
  .8400+01        .5000+01         .9000+00         .1000+00
  .8500+01        .9000+01         .5789+00         .4211+00
  .8600+01        .4000+01         .2872+00         .7128+00
```

Figure 3-10 (Continued).

```
.8700+01    .2000+01    .2688+00    .7312+00
.8800+01    .2000+01    .4022+00    .5978+00
.8900+01    .4000+01    .2088+00    .7912+00
.9000+01    .2000+01    .9400+00    .6001-01
.9100+01    .0000       .9250+00    .7500-01
.9200+01    .0000       .1000+01    .3974-05
.9300+01    .1000+01    .8000+00    .2000+00
.9400+01    .2000+01    .9000+00    .1000+00
.9500+01    .7000+01    .6526+00    .3474+00
.9600+01    .3000+01    .3404+00    .6596+00
.9700+01    .4000+01    .3226+00    .6774+00
.9800+01    .3000+01    .2935+00    .7065+00
.9900+01    .4000+01    .3846+00    .6154+00
.1000+02    .2000+01    .8200+00    .1800+00
.1010+02    .1000+01    .8750+00    .1250+00
.1020+02    .1000+01    .9000+00    .1000+00
.1030+02    .1000+01    .1000+01    .5960-05
.1040+02    .2000+01    .9000+00    .1000+00
.1050+02    .6000+01    .4526+00    .5474+00
.1060+02    .2000+01    .2660+00    .7340+00
.1070+02    .3000+01    .4086+00    .5914+00
.1080+02    .6000+01    .2717+00    .7283+00
.1090+02    .1000+01    .4505+00    .5495+00
.1100+02    .4000+01    .7400+00    .2600+00

 MINIMUM    .0000       .0000       .1397-06
 MAXIMUM    .9000+01    .1000+01    .1000+01
```

Figure 3-10 (Continued).



```
                                   **PLOT NUMBER  1**
                                    RUN  NUMBER  2

                                    SCALES OF PLOT
Q=QUE          .0000          .2500+01          .5000+01          .7500+01          .1000+02
U=USED         .0000          .2500+00          .5000+00          .7500+00          .1000+01
F=FREE         .3000          .2500+00          .5000+00          .7500+00          .1000+01

TIME      0    5   10   15   20   25   30   35   40   45   50   55   60   65   70   75   80   85   90   95  100 DUPLICATES
```

```
OUTPUT   CONSISTS  OF    100 POINT SETS (   300 POINTS)
STORAGE  ALLOCATED FOR  1125 POINT SETS (  4500 WORDS )
STORAGE   NEEDED   FOR   100 POINT SETS (   400 WORDS )
```

Figure 3-11 Plot Output for MUXDA in Second Run.

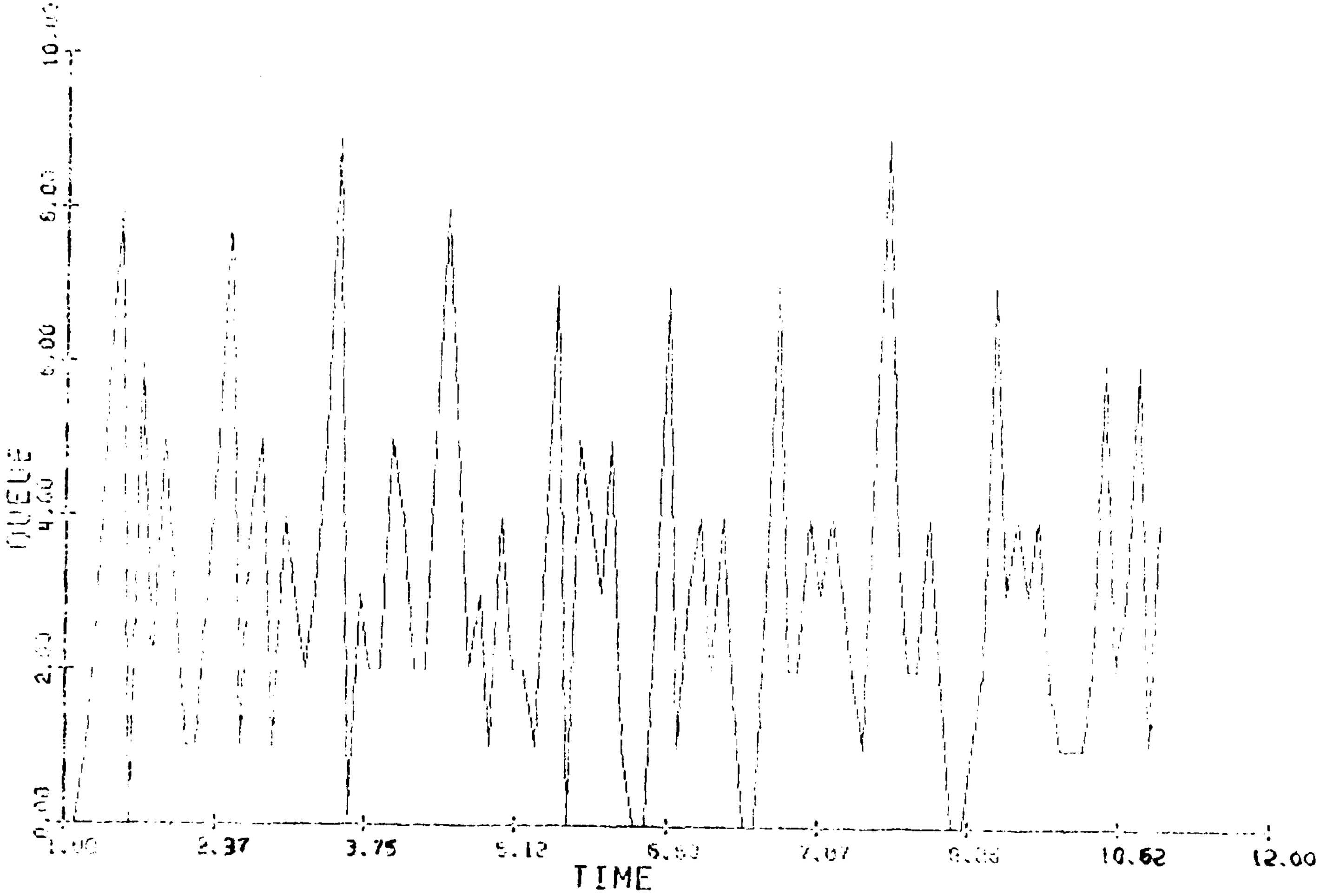


Figure 3-12 Variation of Queue Length with Time in MUXDA (Second Run).

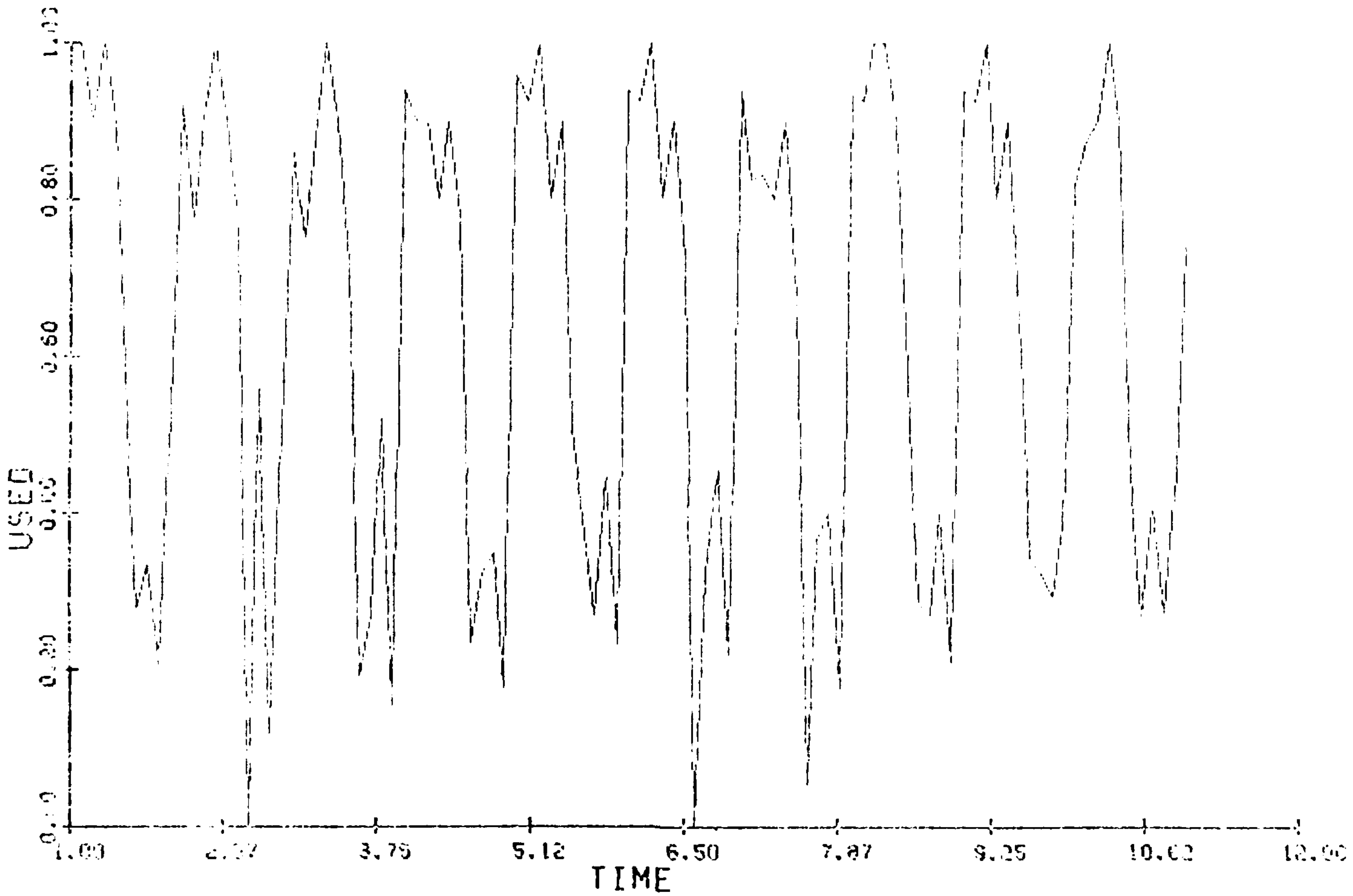


Figure 3-13 Variation of the Percentage of Interval Time Available for Demand Message Transmission with Time in MUXDA (Second Run).

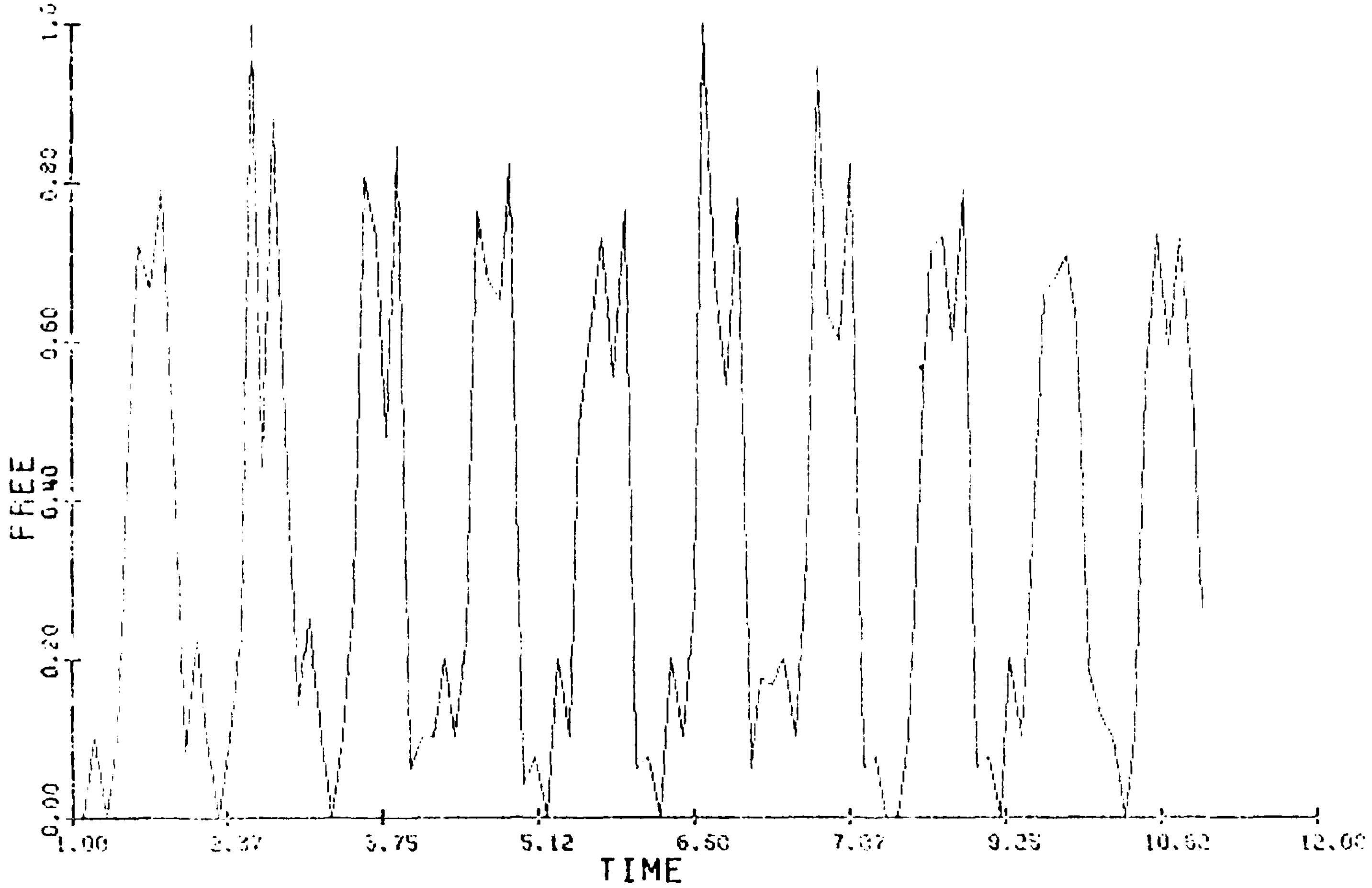


Figure 3-14 Variation of Bus Idle Time with Time in MUXDA (Second Run).

A DESCRIPTION OF THE MUXDB PROGRAM

The MUXDB model refers to redundancy management and fault-handling phases of the multiplex system. Regarding the means by which faults will be dealt with and system redundancy controlled, this is an area of great complexity in the design of multiplex systems. Redundancy is designed into multiplex systems to reduce malfunctions.

MUXDA and MUXDB basically have the same key features, but both models are important because it takes longer to cycle through an equivalent simulated time for MUXDB than it does for MUXDA. MUXDB is a more complex form of model DA; the differences include the following:

(1) extreme noise impact on the data bus

(2) terminal malfunction and impact of failure recovery

(3) the command and response dealt with on an individual message basis

(4) introduction of message transmission time uncertainties

Statement of the Problem

Model DB represents a system consisting of demand access background message queueing and foreground fixed format message transmission. After completing the fixed message requirement, the demand access messages are transmitted on a first-in, first-out basis. Model DB takes into account the impact of noise on the dual redundant data bus. In order for a failure to result, both data buses must be impacted by a noise event during transmission of the same message. In this model, bus failures are generated by using a noise event of

ATTRIBUTES	FILE 1	FILE 2	FILE 3
File Definition	Events	Arrived Demand Messages	Temporary Storage File
ATRIB (1)	Event Time	Time of Arrival	Event Time
(2)	Event Type where Event Type = I*100 + J and I is event type and J is sub-type	1000. + Demand Message Number	Same as File 1
(3)	Message number (for event type 100) Message number plus number of hits on buses (for event types 200 and 300) Noise generator number (for event types 400 or 500) Don't care (for other event types)	Time of Arrival in Waiting Queue	Same as File 1

I = Event Type

100 - Watchdog Timer
200 - Call to a Terminal
300 - Terminal Response to a Call
400 - Noise Start
500 - Noise Stop
600 - Terminal Recovery from Failure
700 - Terminal Failure
800 - Start a FUI
1000 - Demand Message Arrival

J = Subtype

I	J
100 -	Dummy Argument
200 -	Terminal Number 1,NTR
300 -	Terminal Number 1,NTR
400 -	Bus/Noise Designation 1,2,3
500 -	Bus/Noise Designation 1,2,3
600 -	Terminal Number 1,NTR
700 -	Terminal Number 1,NTR
800 -	FUI Number 1,NFUIS
1000 -	Demand Message Number 1,NDM

Table 3-3 Definition of GASP Files.

Variable	Definition
DM (I,J)	Demand Message Parameter Definition. J = Demand Message Number I = Demand Message Parameter where: 1 = Mean time between Demand Message Occurrences 2 = Maximum Δ^* of uniform distribution about Mean 3 = Demand Message length 4 = Terminal to call for this Message
DME	Time of the end of the response of the last demand message sent

FIX (I,J)	Fixed message lengths per FUI
	J = FUI number
	I = (I,NFIX(J)) for particular Fixed Message within FUI

FME	Time of the end of the response of the last fixed message sent
FUI	Fundamental Update Interval (FUI) for data transmission on bus
FUI1T	The time of occurrence of the last FUI numbered 1
FUIPRS	Time of start of actual message or process for FUI
FUISTA	The actual time of FUI start
IBUSY	Bus Controller Busy Transmitting or Awaiting Response (0 = Not Busy, 1 = Busy)
ICURF	The FUI number currently being processed (1,NFUIS)
ILBUS	The current number of noise events disrupting transmission on the left bus (0,NNOISE)

Table 3-4 Definition of Non-GASP Variables.

* Time increment between two successive points.

Variable	Definition

IØK (I)	Terminal Up/Down Status
	I = Terminal Number (1,NTR)
	IØK(I) = (0=Up and Working, 1=Busted)

IRBUS	The current number of noise events disrupting transmission on the right bus (0,NNOISE)
IRESE	The number of valid readible responses lost to timeout
JIT	The switch to detect occurrence of the first scheduled message
MSGNØW	The current fixed message number, in FUI, ICURF, being processed at this time if MSGTYP=1 (Don't care if MSGTYP≠1)
MSGTYP	The current message type being processed by the bus controller. Where MSGTYP equals: 1 for fixed messages 2 for demand messages
NB	Number of noise hits on both buses
NDM	Maximum Quantity of Demand Messages which the System must process
NFIX (I)	The number of fixed messages to be processed by an individual FUI I = Particular FUI Number (1,NFUIS)
NFUIS	Maximum number of FUI intervals. This is the number of minor frame cycles per major frame.
NL	Number of noise hits on left bus
NMSG	Total number of messages sent during simulation
NMSGH	Total number of messages hit on at least one bus
NNØISE	Number of different noise generators impacting the system. This is the number of user input noise generator cards (Max=15).

Table 3-4 Definition of Non-GASP Variables (Continued).

Variable	Definition
NSBUS (I)	Noise generation parameter for generator I which denotes bus(es) impacted. (I=1,NNOISE) and NSBUS(I)=: 1 for left bus affected 2 for right bus affected 3 for both buses affected.
NR	Number of noise hits on right bus
NTM	User input of quantity of terminal down events affecting system. (Note: a terminal down event can affect any terminal in system).
NTO	Number of timeouts
NTP	User input of quantity of terminals in system
NUMV (I)	(Unused variable)
SLNG (I)	Mean length for occurrence of noise event generated by noise generator I. Where I = (1,NNØISE)
SLVAR	Maximum Δ of uniform distribution about length SLNG(I) for occurrence noise generator I where I = (1,NNØISE)
SMEAN (I)	Mean time between occurrences of noise events generated by noise generator I(I = (1,NNØISE)).
SMVAR (I)	Maximum Δ of uniform distribution about SMEAN(I). I is noise generator number (1,NNØISE).
TLEN (I)	The length of a terminal down time for terminal down event generator I. (I = (1,NTM)).
TLVAR (I)	The maximum Δ of uniform distribution about time of terminal down TLEN(I) generated by terminal down generator I. (I=(1,NTM))
TMEAN (I)	Mean time for occurrence of next terminal down by terminal down generator I where I = (1,NTM).
TMSENT (I)	Total message lengths sent per FUI number I where I = (1,NFUIS).

Figure 3-4 Definition of Non-GASP Variables (Continued).

Variable	Definition
TMVAR (I)	Maximum Δ of uniform distribution about TMEAN(I), the time of occurrence of next terminal down by terminal down generator I where I = (1,NTM).
TRES	The mean time of terminal I response where I = (1,NTR).
TRVAR (I)	Maximum Δ of uniform distribution about TRES(I), the response time of terminal I where I = (1,NTR).
XFUI	FUI in subroutine FUI to avoid labeling problem.

Table 3-4 Definition of Non-GASP Variables (Continued).

endless duration. The foreground transmission of a message is sent on a message-by-message basis, instead of a fixed non-varying sequence group. This scheme evaluates the impact of noise on the message. For separate evaluation, the message is separated into the command segment and the response segment. Failure can be recognized by either a non-responding terminal or a failure of the controller to acknowledge the response. A failure can also be determined by a "watch-dog" timer event that occurs before a given message response.

The occurrence of a failure event causes each terminal on the bus to fail. There are two failure modes: (1) permanent disable or (2) intermittent disable (that is, a terminal which recovers from a failure after a period of time). There is a response time associated with each message, which is uniformly distributed. Therefore, under this situation a controller cannot predict the length of time for transmitting a message. If the length of time for an average message transmission is less than the time to the start of the next FUI time, the controller will schedule its occurrence; however, this program has a built in feature which allows a delay of the next FUI fixed format message transmission until the transmission in progress is completed.

Simulation Objectives

The objectives of this simulation are to determine bus resource utilization for this approach of data handling, in addition to obtaining a measure of the impact of bus uncertainties on the data transportation timing. The bus uncertainties include variable response delay, bus noise and corresponding watch dog requirements,

AD-A097 375   MISSISSIPPI STATE UNIV MISSISSIPPI STATE ENGINEERING--ETC F/G 9/2
A STUDY OF AVIONICS TIME DIVISION MULTIPLEX BUS SIMULATION.(U)
DEC 80 M D CALHOUN, B KHATIRZAD   AFOSR-80-0126
UNCLASSIFIED   MSSU-EIRS-EE-81-1   AFOSR-TR-81-0311   NL

2 of 2
AD
A097375

END
DATE
FILMED
5 81
DTIC

and terminal failure and corresponding watch-dog requirements.

GASP IV Simulation Structure and Program Variables for MUXDB

In this program, there are three files used. Table 3-3 gives the definitions of the files and their associated characteristics for this simulation. File 1 is the event file as per GASP IV, File 2 stores the demand message arrival, and File 3 is the temporary storage file. Table 3-4 defines the non-GASP variables.

Main Program, Subroutine INTLC, and Subroutine EVNTS Description

Main Program

The main program sets the card reader number (NCRDR) and the card printer number (NPRNT) and subroutine GASP is called. The MUXSIM executive system is not used, therefore it is not necessary to use subroutines CHAIN, RESTOR, and GETCOM.

Subroutine INTLC

This subroutine is called via subroutine DATIN, in order to read in the simulation data cards and to set up the initial conditions from the input data cards or algebraic statements. The non-GASP user input data are printed out to make checking easier.

Subroutine EVNTS

Subroutine EVNTS sends control to one of the nine user written subroutines: WATCH, CTERM, TERMR, NOISES, NOISET, TRMUP, TRMDN, FUI, and DMARIV. The events of the simulation, in the order of their event code are:

100-Watch-Dog Timer (WATCH)

200-Call to Terminal (CTERM)

300-Terminal Response to a Call (TERMR)

400-Noise Start (NOISES)

500-Noise Stop (NOISET)

600-Terminal Recovery from Failure (TRMUP)

700-Terminal Failure (TRMDN)

800-Start a Fundamental Update Interval (FUI)

1000-Demand Message Arrival (DMARIV)

Subroutine WATCH (IDUM)

The "watch-dog" timer event is established in subroutine WATCH, which performs the following functions:

(1) The program increases the number of timeouts (NTO) by one.

(2) To set the requirement for testing a next message, it calls subroutine NXTMSG (2).

Subroutine CTERM (ITRM)

The call to terminal events is accomplished in subroutine CTERM, which performs the following functions:

(1) This routine tests for noise hits on the data bus.

(2) CTERM handles message arrival and processing at a terminal and produces terminal response.

(3) CTERM prevents response if noise hits on both buses or if the terminal is down.

(4) If there is no response inhibit, CTERM schedules a response.

(5) CTERM increases the number of messages by one in order to record

the total number of messages sent during the simulation.

Subroutine TERMR (ITRM)

Terminal response is accomplished in subroutine TERMR, which performs the following functions:

(1) This routine checks to see if the terminal response is valid; if it is not, the "watch-dog" timer is left alone.

(2) It illiminates the "watch-dog" timer and calls the next message (NXTMSG (3)) subroutine to program the following message if the terminal response is valid.

(3) If the terminal response is valid, but the "watch-dog" timer has expired, IRESE is increased by one.

Subroutine NOISES (IN)

The noise event process is accomplished in subroutine NOISES, which performs the following functions:

(1) NOISES processes noise events and schedules the next noise event arrival and duration.

(2) It establishes which buses are impacted.

(3) This routine marks the message on the bus hit according to noise.

(4) It increases the count of number of noise events on the proper buses by one.

(5) NOISES establishes the end-of-the-noise event.

(6) It calls the STAT subroutine to record the bus noise data.

Subroutine NOISET (IN)

Noise event termination is established in subroutine NOISET, which performs the following functions:

(1) This routine reduces the number of noise events on the proper buses by one to record the number of noise events left.

(2) To record the bus noise statistics, it calls subroutine STAT.

Subroutine TRMUP (ITN)

Terminal recovery from failure is established in subroutine TRMUP, which performs only one function: it decreases the terminal up/down status IOK(IN) by one. The terminal is operational if the indicator is zero.

Subroutine TRMDN (ITN)

Terminal failure is established in subroutine TRMDN, which performs the following functions:

(1) To indicate that the terminal is down, it sets the up/down status IOK(ITN) up one.

(2) TRMDN schedules the next down event for this terminal.

Subroutine FUI (IFUI)

The start of the fundamental update interval is established in subroutine FUI, which performs the following functions:

(1) If FUI = 1, it proceeds to compute the time of the next FUI (1) start to prevent the round-off error from accumulating and invalidating the results. It then produces the entire schedule for all FUI starts for the remainder of this frame and schedules the arrival of the FUI (1) for the next major frame. If FUI ≠ 1, it omits the above and starts at this point.

(2) It sets the value for current FUI (ICURF) and sets the value of the fixed message number to be transmitted to 1, the message type to 1 to indicate a fixed message, and the JTT switch to I to enable the detection of the first message to be transmitted. It then calls subroutine NXTMSG (1) to schedule the next possible call to a terminal.

Subroutine DMARIV (IDM)

The arrival of a demand message is established in subroutine DMARIV, which performs the following functions:

(1) DMARIV establishes the length, message number, and arrival time.

(2) It files this information in File (2) or the arrived demand message file.

(3) It schedules the next arrival of this demand message.

Subroutine OTPUT

Subroutine OTPUT is used to gain output in addition to the standard GASP IV summary report. OTPUT is called prior to subroutine SUMRY and is used to print out the following:

(1) The number of timeouts or intervals of time between bus failure and bus recovery.

(2) The number of valid readable responses lost to timeouts.

(3) The number of bits on the left bus, right bus, and on both buses.

(4) The total number of messages sent.

(5) The total number of messages to hit on one bus or more.

Subroutine NXTMSG (IAM)

The subsequent call to a terminal scheduling is established in subroutine NXTMSG, which performs the following functions:

(1) If it is the start of FUI and the bus is busy either waiting on transmission or a transmission is in progress, it returns to the subroutines that have been called.

(2) NXTMSG branches on message type to 3 or 5.

(3) It collects a histogram and statistics information for the first fixed message in FUI.

(4) If possible, it schedules a fixeu message transmission and a companion "watch-dog" timer event. It then updates the message number by one, sets the IBUSY flag to busy (one) and increases the message count by one, and returns.

(5) For a demand message, NXTMSG tests to see if there is time to send a demand message; if there is, it schedules a call to a terminal and schedules a companion "watch-dog" terminal event, sets the busy flag, and increases the message count by one.

(6) If a demand message cannot be scheduled due to lack of time to achieve the transmission, this routine sets the busy flag to free and establishes the end of demand message transmission for this FUI. It then returns to the subroutines that have been called.

Subroutine STAT (ILBUS, IRBUS)

Statistics of time-persistent variables for noise on the left bus, right bus, and/or both are collected for noise events on the bus; this is the only function that subroutine STAT performs.

Function RNXT (RMEN, RVAR, ISTRM)

This routine computes the delta time for the next event arrival for a given event generation, which is based on a uniform distribution about the mean, using the random number stream designated by the user. RNXT performs the following functions:

(1) For calling the next arrival of the event in question, RNXT establishes the random number stream.

(2) For the event in question, it establishes the uniform upper and lower bound.

(3) By using the uniform distribution, RNXT computes the arrival time.

Simulation Report

The MUXDB outputs are given in the following section, with particular emphasis on some of the more common data outputs:

Figure 3-15 shows the input data echo check, provided by subroutine DATIN. Figure 3-16 presents the user input cards; Figure 3-17 is similar to Figure 3-16, but is given in more detail. The printout of the files that are obtained at the end of subroutine DATIN (Time 0) is shown in Figure 3-18, and a partial printout of the event tracing which is obtained from subroutine MONTR is shown in Figure 3-19.

Figure 3-20 is the GASP IV summary report. The statistics that are collected using subroutine COLCT are presented; these show that the interval time between the start of an actual message and the response of the last demand message is, on the average, .016 of the time unit with the standard deviation of .0026 of the time unit. These values are based on 1,000 observations. The minimum and maximum values observed for the time interval between the start of an actual message and the end of the response of the fixed message are .011 and .022 of the time unit, respectively. The average interval time between the start of an actual message and the end of the last demand message is .088 of the time unit, based on 1,000 observations.

The next set of statistics on the summary report is for the data collected in subroutine TIMST; this shows that the utilization of noises on the right bus is .018 of the time unit over the total simulation time of 1,000 of the time unit.

Statistics regarding the use of files are shown in Figure 3-21. For example, on the average, there are 23.3 events in file 1. A maximum of 32 events are stored in the event file.

In Figure 3-22, statistical information for the fixed message in FUI is given. The figure shows that 91.4% of the first fixed messages, based on 10,001 observations, has the starting time less than or equal to .014 of the time unit.

```
                                          SIMULATION PROJECT NUMBER    5  BY    BEHROOZ

                                          DATE  4/ 21/   80          RUN NUMBER     1 OF      1
                                          LLSUP=0000000000000000    GASP IV VERSION 18MAY74

NNCLT=        2     NNSTA=        3     NNHIS=      1     NNPRM=       0     NNPLT=       0     NNSTR=       5     NNTRY=    100
NNATR=        3     NNFIL=        3     NNSET=   5000     NNEQD=       0     NNEQS=       0     NFLAG=       0

COLCT NO.   1     LLABC=IFFL
COLCT NO.   2     LLABC=ITFL

TIMST NO.   1     LLABT=NRTBUS      I.C. =         .0000
TIMST NO.   2     LLABT=NLTBUS      I.C. =         .0000
TIMST NO.   3     LLABT=NBTBUS      I.C. =         .0000

HISTO NO.   1     LLABH=JIT                                NNCEL=      10     HHLOW=          .0000     HHWID=          .2000-02

KKRNK= (    0)        0        0

IINN = (    3)        3        3

MSTOP=        1     JJCLR=        1     JJBEG=      1     IICRD=       0     TTBEG=          .0000     TTFIN=          .1000+04
JJFIL=        1
IISED= 65397              43163             19249              45212             15893
```

Figure 3-15 GASP IV Input Data Echo Check for MUXDB Simulation.

```
          INPUT CARDS
C
L        10    .1
D              .1          .003         .006          1.
D              .3          .004         .005          1.
D              .1          .004         .005          1.
D              .2          .003         .006          2.
D              .1          .003         .006          2.
D              .3          .003         .005          3.
D              .3          .003         .004          3.
D              .3          .003         .006          3.
D              .2          .003         .006          1.
C
R         1    .01         .005
K         2    .01         .003
F         1  .005
F         2  .005
F         3  .005
F         4  .006
F         5  .006
F         6  .006
F         7  .006
F         8  .006
F         9  .006
F        10  .006
N         1       10.00        0.05         .05          .001
N         3       15.0         0.05         .05          .001
N         2       10.00        0.05         .05          .001
N         1       10.00        0.05         .05          .001
N         1       20.0         0.05         .05          .001
N         2       15.0         0.05         .05          .001
N         3       25.0         0.05         .05          .001
T         1        0.5        0.010      1100.0   10.  0
T         2        0.5        0.010      1100.0   10.  0
X
```

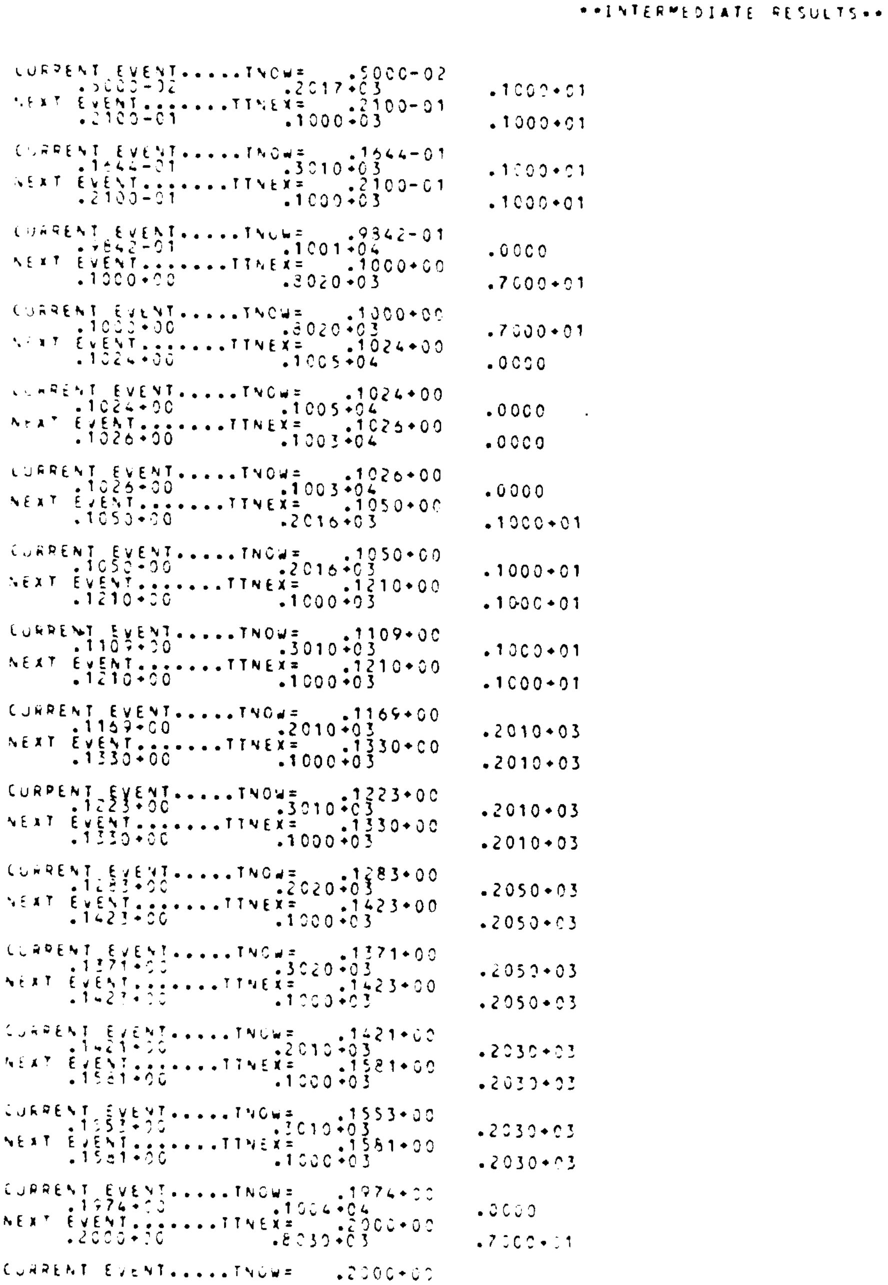
Figure 3-16 User Input Data for GASP IV in MUXDB Simulation.

```
                                  DEMAND MESSAGES

MEAN                 VARIANCE               LENGTH                  TERMINAL NO.
     .1000                .0010                  .0060                  1.0000
     .3000                .0040                  .0050                  1.0000
     .1000                .0040                  .0050                  1.0000
     .2000                .0030                  .0060                  2.0000
     .1000                .0030                  .0060                  2.0000
     .3000                .0030                  .0050                  3.0000
     .3000                .0030                  .0040                  3.0000
     .3000                .0030                  .0060                  3.0000
     .2000                .0030                  .0060                  1.0000


FUI - NO. FIXED MESSAGES - MESSAGE LENGTHS

 1   1        .0050
 2   1        .0050
 3   1        .0050
 4   1        .0060
 5   1        .0060
 6   1        .0060
 7   1        .0060
 8   1        .0060
 9   1        .0060
10   1        .0060


                                   TERMINAL FAULT PARAMETERS

TERM NO             MEAN                 VARIANCE             LENGTH               VARIANCE OF LENGTH
    1                    .5000               .0100            1100.0000               10.0000
    2                    .5000               .0100            1100.0000               10.0000


                                   TERMINAL RESPONSE

TERM NO             RESPONSE             VARIANCE OF RESPONSE TIME
    1                    .0100               .0050
    2                    .0100               .0030


                                   NOISE PARAMETERS

BUS CODE            MEAN                 VARIANCE             LENGTH               VARIANCE OF LENGTH
    1                 10.0000                .0500                .0500                .0010
    3                 15.0000                .0500                .0500                .0010
    2                 10.0000                .0500                .0500                .0010
    1                 10.0000                .0500                .0500                .0010
    1                 20.0000                .0500                .0500                .0010
    2                 15.0000                .0500                .0500                .0010
    3                 25.0000                .0500                .0500                .0010

NFUIS =    10 FUI=     .10000
```

Figure 3-17 User Detailed Input for MUXDB.

```
**GASP FILE STORAGE AREA DUMP AT TIME      .0000   **

MAXIMUM NUMBER OF ENTRIES IN FILE STORAGE AREA = 21

                                   PRINTOUT OF FILE NUMBER    1
                                        TNOW =      .0000
                                        QQTIM=      .0000

                                           FILE CONTENTS
ENTRY   1    =      .0000       .8010+03       .7300+01
ENTRY   2    =      .0000       .0000          .7000+01
ENTRY   3    =      .9842-01    .1001+04       .0000
ENTRY   4    =      .1024+00    .1005+04       .0000
ENTRY   5    =      .1026+00    .1003+04       .0000
ENTRY   6    =      .1974+00    .1004+04       .0000
ENTRY   7    =      .2022+00    .1009+04       .0000
ENTRY   8    =      .2972+00    .1008+04       .0000
ENTRY   9    =      .2980+00    .1006+04       .0000
ENTRY  10    =      .3004+00    .1007+04       .0000
ENTRY  11    =      .3039+00    .1002+04       .0000
ENTRY  12    =      .5018+00    .6020+03       .0000
ENTRY  13    =      .5088+00    .6010+03       .0000
ENTRY  14    =      .1100+01    .0000          .0000
ENTRY  15    =      .9973+01    .4010+03       .7000+01
ENTRY  16    =      .9997+01    .4010+03       .4000+01
ENTRY  17    =      .1002+02    .4020+03       .1000+01
ENTRY  18    =      .1497+02    .4020+03       .3000+01
ENTRY  19    =      .1500+02    .4030+03       .6000+01
ENTRY  20    =      .1999+02    .4010+03       .2000+01
ENTRY  21    =      .2496+02    .4030+03       .5000+01
                                               .7000+01

                                   PRINTOUT OF FILE NUMBER    2
                                        TNOW =      .0000
                                        QQTIM=      .0000

                                        THE FILE IS EMPTY

                                   PRINTOUT OF FILE NUMBER    3
                                        TNOW =      .0000
                                        QQTIM=      .0000

                                        THE FILE IS EMPTY
```

Figure 3-18 Printout of Files at Time 0 for MUXDB.



```
                                                            **INTERMEDIATE RESULTS**

CURRENT EVENT.....TNOW=    .5000-02
     .5000-02           .2017+03          .1000+01
NEXT EVENT.......TTNEX=    .2100-01
     .2100-01           .1000+03          .1000+01

CURRENT EVENT.....TNOW=    .1644-01
     .1644-01           .3010+03          .1000+01
NEXT EVENT.......TTNEX=    .2100-01
     .2100-01           .1000+03          .1000+01

CURRENT EVENT.....TNOW=    .9842-01
     .9842-01           .1001+04          .0000
NEXT EVENT.......TTNEX=    .1000+00
     .1000+00           .8020+03          .7000+01

CURRENT EVENT.....TNOW=    .1000+00
     .1000+00           .8020+03          .7000+01
NEXT EVENT.......TTNEX=    .1024+00
     .1024+00           .1005+04          .0000

CURRENT EVENT.....TNOW=    .1024+00
     .1024+00           .1005+04          .0000
NEXT EVENT.......TTNEX=    .1026+00
     .1026+00           .1003+04          .0000

CURRENT EVENT.....TNOW=    .1026+00
     .1026+00           .1003+04          .0000
NEXT EVENT.......TTNEX=    .1050+00
     .1050+00           .2016+03          .1000+01

CURRENT EVENT.....TNOW=    .1050+00
     .1050+00           .2016+03          .1000+01
NEXT EVENT.......TTNEX=    .1210+00
     .1210+00           .1000+03          .1000+01

CURRENT EVENT.....TNOW=    .1109+00
     .1109+00           .3010+03          .1000+01
NEXT EVENT.......TTNEX=    .1210+00
     .1210+00           .1000+03          .1000+01

CURRENT EVENT.....TNOW=    .1169+00
     .1169+00           .2010+03          .2010+03
NEXT EVENT.......TTNEX=    .1330+00
     .1330+00           .1000+03          .2010+03

CURPENT EVENT.....TNOW=    .1223+00
     .1223+00           .3010+03          .2010+03
NEXT EVENT.......TTNEX=    .1330+00
     .1330+00           .1000+03          .2010+03

CURRENT EVENT.....TNOW=    .1283+00
     .1283+00           .2020+03          .2050+03
NEXT EVENT.......TTNEX=    .1423+00
     .1423+00           .1000+03          .2050+03

CURRENT EVENT.....TNOW=    .1371+00
     .1371+00           .3020+03          .2050+03
NEXT EVENT.......TTNEX=    .1423+00
     .1423+00           .1000+03          .2050+03

CURRENT EVENT.....TNOW=    .1421+00
     .1421+00           .2010+03          .2030+03
NEXT EVENT.......TTNEX=    .1581+00
     .1581+00           .1000+03          .2030+03

CURRENT EVENT.....TNOW=    .1553+00
     .1553+00           .3010+03          .2030+03
NEXT EVENT.......TTNEX=    .1581+00
     .1581+00           .1000+03          .2030+03

CURRENT EVENT.....TNOW=    .1974+00
     .1974+00           .1004+04          .0000
NEXT EVENT.......TTNEX=    .2000+00
     .2000+00           .8030+03          .7000+01

CURRENT EVENT.....TNOW=    .2000+00
```

Figure 3-19 Output from Subroutine MONTR, showing Event Tracing for MUXDB.

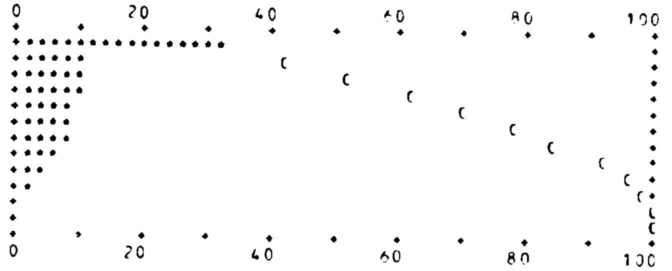
```
NUMBER OF VALID READABLE RESPONSES LOST TO TIMEOUT =          0

NUMBER OF TIMEOUTS =          0

NUMBER OF HITS ON LEFT BUS =       248 RIGHT BUS =       166 BOTH BUSES =        211

TOTAL NUMBER OF MESSAGES SENT =    126834

TOTAL NUMBER OF MESSAGES HIT ON AT LEAST ONE BUS =      1147


                                        **GASP SUMMARY REPORT**

                                   SIMULATION PROJECT NUMBER    5  BY    BEHROOZ

                                   DATE  4/ 21/   80        RUN NUMBER     1 OF     1

                                   CURRENT TIME =     .1000+04


                                  **STATISTICS FOR VARIABLES BASED ON OBSERVATION**
              MEAN            STD DEV          SD OF MEAN          CV              MINIMUM            MAXIMUM        OBS

IFFL        .1620-01         .2558-02          .8089-04          .1579+00          .1100-01           .2205-01      1000
ITFL        .8836-01         .2553-01          .8072-03          .2889+00          .1133-01           .1704+00      1000

                                    **STATISTICS FOR TIME-PERSISTENT VARIABLES**
              MEAN            STD DEV          MINIMUM           MAXIMUM          TIME INTERVAL     CUR. VALUE

NRTBUS      .1765-01         .1449+00          .0000             .4000+01          .1000+04          .0000
NLTBUS      .1354-01         .1238+00          .0000             .3000+01          .1000+04          .1000+01
NBTBUS      .7329-02         .8833-01          .0000             .3000+01          .1000+04          .0000
```

Figure 3-20 GASP Summary Report for MUXDB.

```
**GASP FILE STORAGE AREA DUMP AT TIME     .1000+04**

MAXIMUM NUMBER OF ENTRIES IN FILE STORAGE AREA   38

                                         PRINTOUT OF FILE NUMBER    1
                                             INOW =     .1000+04
                                             QQTIM=     .1000+04

                                     TIME PERIOD FOR STATISTICS     .1000+04
                                     AVERAGE NUMBER IN FILE          25.2924
                                     STANDARD DEVIATION               2.9932
                                     MAXIMUM NUMBER IN FILE          32

                                                 FILE CONTENTS
ENTRY    1    =       .1000+04       .2021+01           .1000+06
ENTRY    2    =       .1000+04       .5000+03           .3000+01
ENTRY    3    =       .1000+04       .1000+03           .1000+01
ENTRY    4    =       .1000+04       .1001+04           .9999+03
ENTRY    5    =       .1000+04       .1003+04           .9999+03
ENTRY    6    =       .1000+04       .1004+04           .9998+03
ENTRY    7    =       .1000+04       .1002+04           .9997+03
ENTRY    8    =       .1000+04       .4010+03           .1000+01
ENTRY    9    =       .1000+04       .1005+04           .1000+04
ENTRY   10    =       .1000+04       .4010+03           .5000+01
ENTRY   11    =       .1000+04       .4030+03           .7000+01
ENTRY   12    =       .1000+04       .8020+03           .7000+01
ENTRY   13    =       .1000+04       .1009+04           .9999+03
ENTRY   14    =       .1000+04       .1007+04           .9999+03
ENTRY   15    =       .1000+04       .8030+03           .7000+01
ENTRY   16    =       .1000+04       .1008+04           .9999+03
ENTRY   17    =       .1000+04       .1006+04           .9999+03
ENTRY   18    =       .1000+04       .8040+03           .7000+01
ENTRY   19    =       .1000+04       .8050+03           .7000+01
ENTRY   20    =       .1000+04       .8060+03           .7000+01
ENTRY   21    =       .1001+04       .8070+03           .7000+01
ENTRY   22    =       .1001+04       .8080+03           .7000+01
ENTRY   23    =       .1001+04       .8090+03           .7000+01
ENTRY   24    =       .1001+04       .8100+03           .7000+01
ENTRY   25    =       .1001+04       .8010+03           .7000+01
ENTRY   26    =       .1005+04       .4020+03           .6000+01
ENTRY   27    =       .1005+04       .4030+03           .2000+01
ENTRY   28    =       .1010+04       .4010+03           .4000+01
ENTRY   29    =       .1010+04       .4020+03           .3000+01

                                         PRINTOUT OF FILE NUMBER    2
                                             INOW =     .1000+04
                                             QQTIM=     .1000+04

                                     TIME PERIOD FOR STATISTICS     .1000+04
                                     AVERAGE NUMBER IN FILE           2.4248
                                     STANDARD DEVIATION               1.6519
                                     MAXIMUM NUMBER IN FILE          10

                                                 FILE CONTENTS
ENTRY    1    =       .5000-02       .3000+01           .9999+03
ENTRY    2    =       .6000-02       .5000+01           .1000+04

                                         PRINTOUT OF FILE NUMBER    3
                                             INOW =     .1000+04
                                             QQTIM=     .9996+03

                                     TIME PERIOD FOR STATISTICS     .1000+04
                                     AVERAGE NUMBER IN FILE            .0000
                                     STANDARD DEVIATION                .0000
                                     MAXIMUM NUMBER IN FILE           2

                                         THE FILE IS EMPTY
```

Figure 3-21 Statistics Regarding the Use of Files for MUXDB.

```
                                **HISTOGRAM NUMBER  1**

                                          JIT

OBSV     RELA     CUML        UPPER
FREQ     FREQ     FREQ     CELL LIMIT

3145     .314     .314       .0000
1054     .105     .420       .2000-02
 990     .099     .519       .4000-02
 917     .092     .611       .6000-02
 812     .081     .692       .8000-02
 826     .083     .774       .1000-01
 729     .073     .847       .1200-01
 667     .067     .914       .1400-01
 471     .047     .961       .1600-01
 273     .027     .988       .1800-01
  99     .010     .998       .2000-01
  18     .002    1.000        INF
 ---
10001
```



```
*FIN
```

Figure 3-22 Histogram Output for MUXDB.

CHAPTER IV

A SIMULATION EXAMPLE IN VARIOUS LANGUAGES

In this chapter, a single queue, single server simulation example is discussed in four languages; they are: FORTRAN, GASP IV, GPSS II, and SIMSCRIPT II.

The primary goal of this program is to simulate a single queue, single server system. In this thesis, the server is analogous to the terminal or bus controller and the queue is the message queue stored in the controller. The service unit is the data bus itself. The arrival of a message on a bus is exponentially distributed with mean time of five minutes and the service time is also exponentially distributed with mean time of four minutes. This program is simulated for eight hours (simulation time). The second goal is to use FORTRAN to implement the model to realize impropriety of the language for simulation. Figure 4-1 shows the general structure of this problem in the FORTRAN language.

A Description of the FORTRAN Simulation Program

The FORTRAN Simulation Program performs the following functions:

(1) FORTRAN initializes all variables and parameters and sets the total time of simulation.

(2) This program checks to see whether the simulation time is over; if it is over, it calls the proper event to be executed.

(3) FORTRAN calculates the average queue length and the average service time for all runs.

The flow chart of subroutines ARRIVAL, SERVICE, and DEPARTURE is given in Figures 4-2, 4-3, and 4-4, respectively. The printout of this program is given in Figure 4-5.

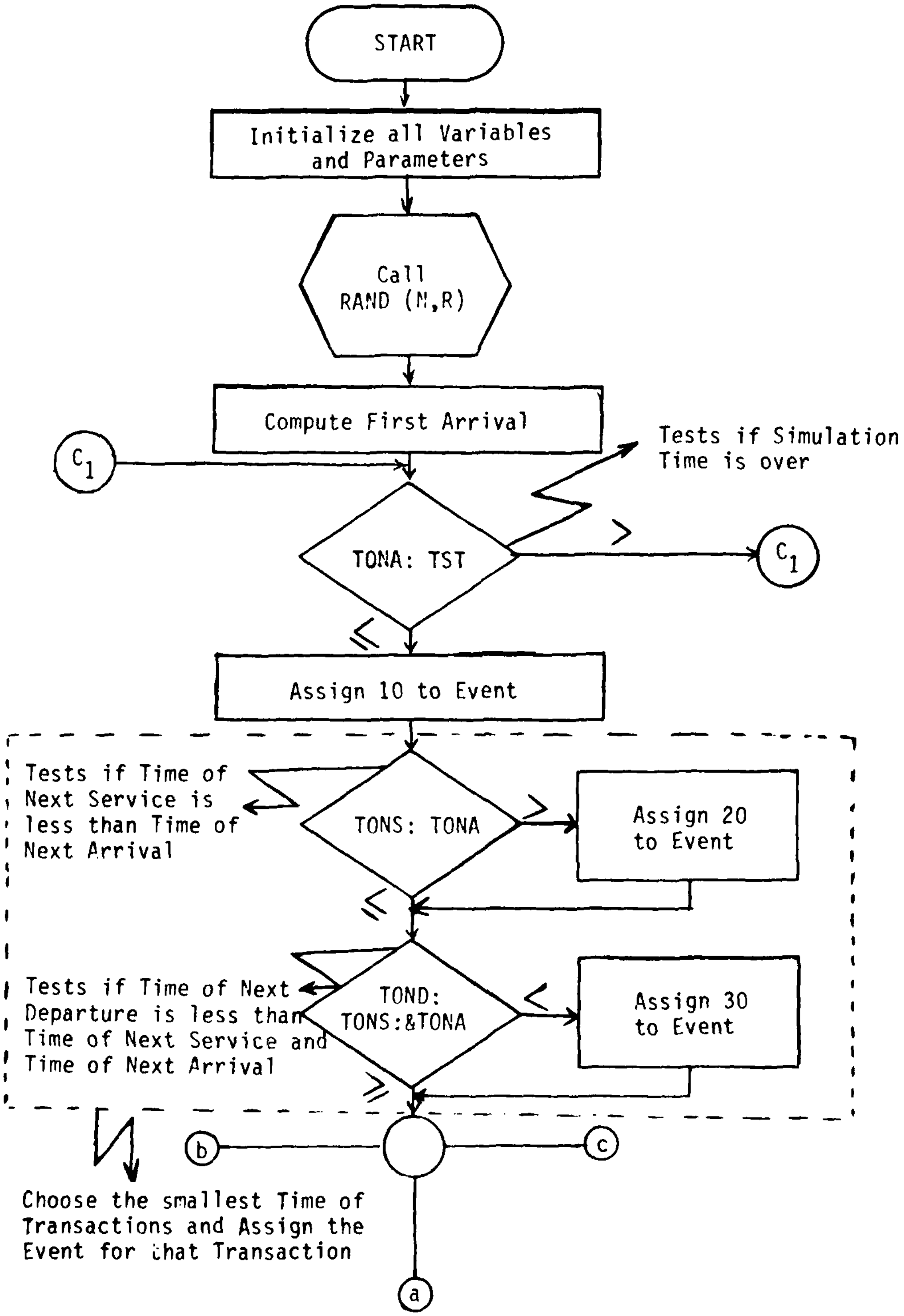

START
Initialize all Variables and Parameters
Call RAND (N,R)
Compute First Arrival
C1
Tests if Simulation Time is over
TONA: TST
>
C1
<
Assign 10 to Event
Tests if Time of Next Service is less than Time of Next Arrival
TONS: TONA
>
Assign 20 to Event
<
Tests if Time of Next Departure is less than Time of Next Service and Time of Next Arrival
TOND: TONS:&TONA
<
Assign 30 to Event
>
b
c
Choose the smallest Time of Transactions and Assign the Event for that Transaction
a

Figure 4-1 Flow Chart of Single Queue, Single Server in the FORTRAN Program.

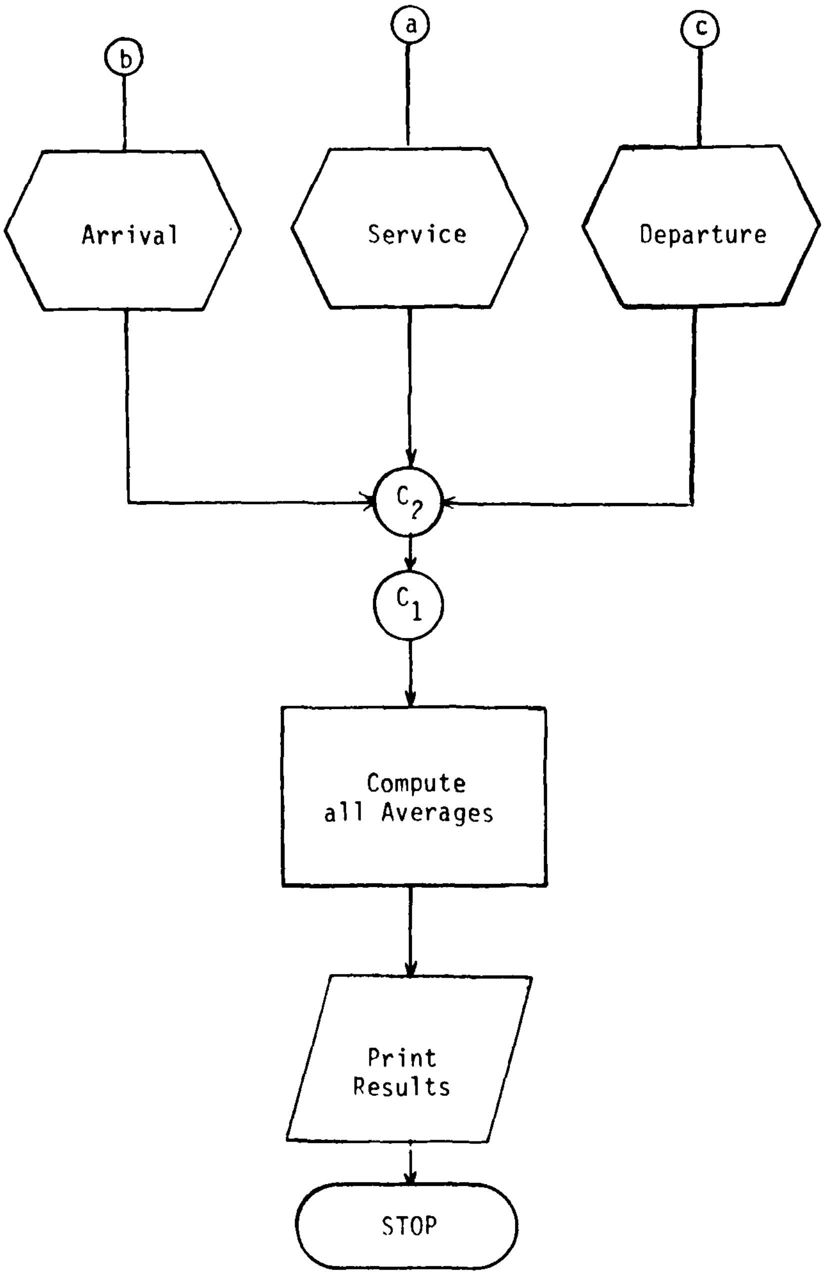


Figure 4-1 Flow Chart of Single Queue, Single Server in the FORTRAN Program (Continued).

Subroutine ARRIVAL

The parameters received by this subroutine are: Time of Next Arrival (TONA), Time of Next Service (TONS), Clock, Time of Last Queue Change (TOLQC), Queue Length (IQ), Total Queue Length (TIQ), Facility Status (IFS), Sum of Queue Length (SOQL), Expected Arrival Time (EXPA), and Seed for Random Number Generator (N).

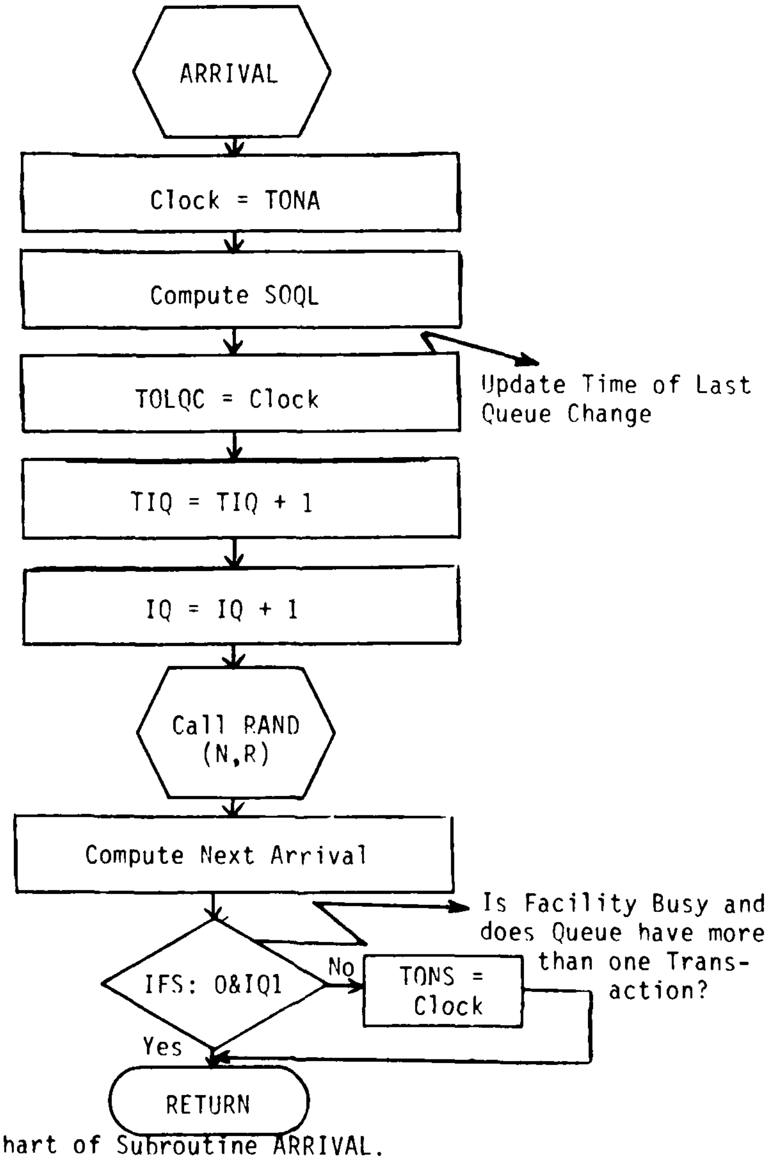


Figure 4-2 Flow Chart of Subroutine ARRIVAL.

Subroutine SERVICE

The parameters received by this subroutine are: Time of Next Service (TONS), Time of Next Departure (TOND), Clock, Time of Last Queue Change (TOLQC), Queue Length (IQ), Sum of Queue Length (SOQL), Sum of Service Time (SOST), Expected Service Time (EXPS), and Seed for Random Number Generator (N).

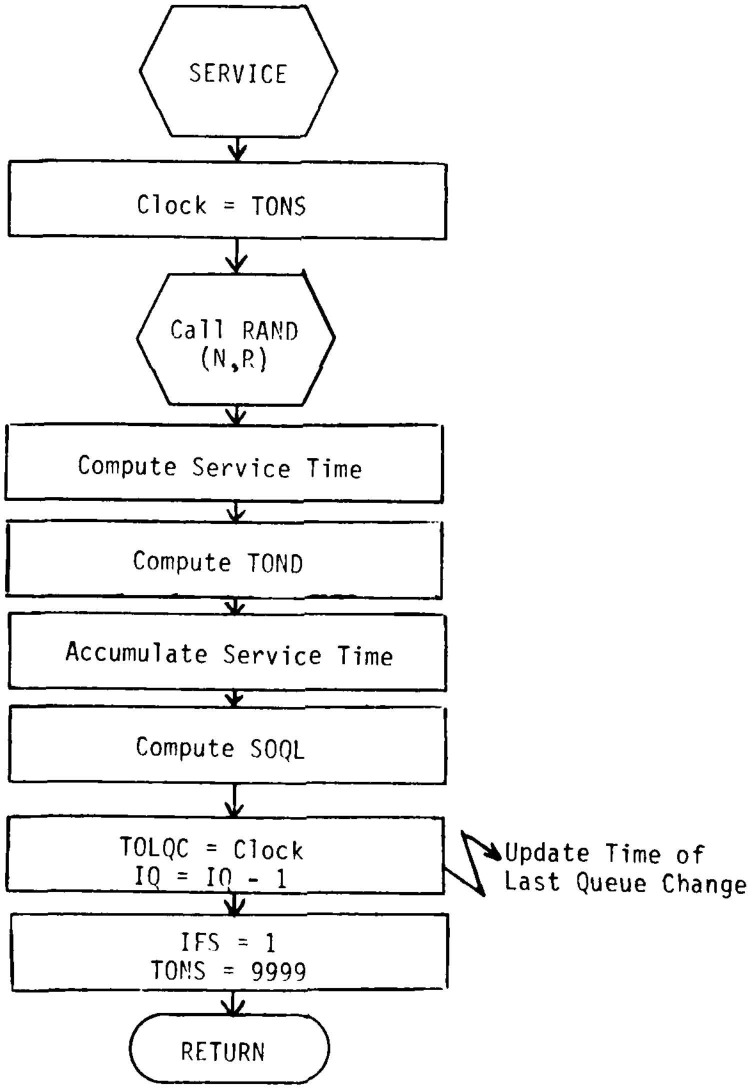


Figure 4-3 Flow Chart of Subroutine SERVICE.

Subroutine DEPARTURE

The parameters received by this subroutine are: Time of Next Departure (TOND), Time of Next Service (TONS), Clock, Queue Length (IQ), and Facility Status (IFS).

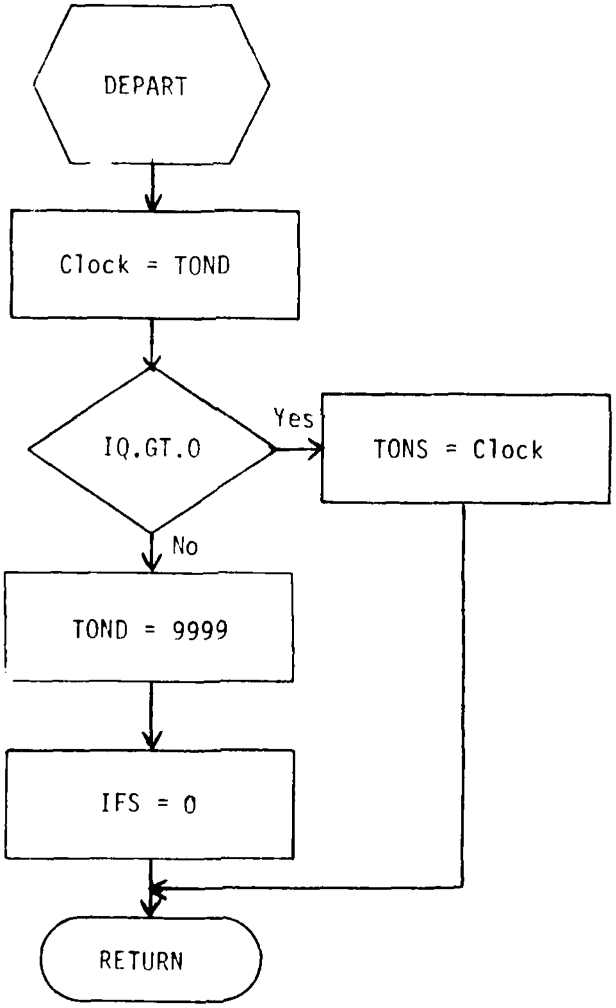
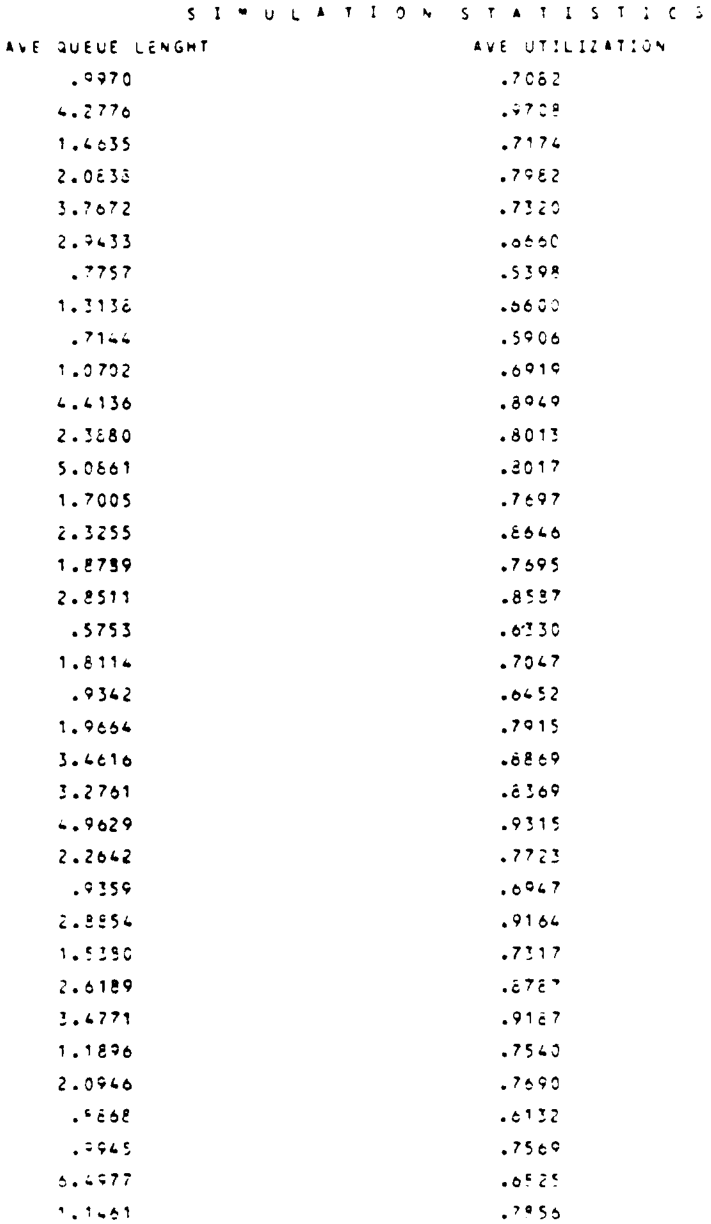


Figure 4-4 Flow Chart of Subroutine DEPARTURE.



```
                     SIMULATION STATISTICS
AVE QUEUE LENGHT                          AVE UTILIZATION
      .9970                                   .7062
     4.2776                                   .9708
     1.4635                                   .7174
     2.0833                                   .7982
     3.7672                                   .7320
     2.9433                                   .6660
      .7757                                   .5398
     1.3136                                   .6600
      .7144                                   .5906
     1.0702                                   .6919
     4.4136                                   .8949
     2.3880                                   .8013
     5.0861                                   .8017
     1.7005                                   .7697
     2.3255                                   .8646
     1.8759                                   .7695
     2.8511                                   .8587
      .5753                                   .6330
     1.8114                                   .7047
      .9342                                   .6452
     1.9664                                   .7915
     3.4616                                   .8869
     3.2761                                   .8369
     4.9629                                   .9315
     2.2642                                   .7723
      .9359                                   .6947
     2.8954                                   .9164
     1.5390                                   .7317
     2.6189                                   .8787
     3.4771                                   .9187
     1.1896                                   .7540
     2.0946                                   .7690
      .5868                                   .6132
      .7945                                   .7569
     6.4977                                   .6525
     1.1461                                   .7956
```

Figure 4-5 Printout of the FORTRAN Simulation Program.

```
  1.3156          .6812
   .9018          .6884
 17.2648          .9922
  1.5735          .7069
  3.5791          .6353
  2.7486          .6363
  1.0270          .6779
  2.8770          .8178
  1.3752          .6719
   .8362          .6226
 10.1216          .9340
  3.2814          .9179
  1.5017          .7428
  5.0549          .9389
  2.1151          .7873
  3.6910          .8481
  1.1818          .6756
  1.6242          .7386
  3.8657          .8750
 10.3328          .5785
   .7718          .7332
  1.1313          .6828
  8.8800          .6844
  1.6529          .7568
  1.8291          .8015
  1.5557          .7985
  3.6046          .8579
  2.3624          .6465
   .5565          .5564
  3.4269          .9500
  1.3281          .7119
  1.6151          .7598
   .5944          .6256
  1.5605          .7602
  1.1294          .7944
  1.0748          .6635
  1.1469          .7652
   .5041          .6219
  1.0667          .7234
  1.5097          .6535
```

Figure 4-5 (Continued).

```
  .7096                                   .7539
 1.3979                                   .7234
 1.9040                                   .7235
 4.6719                                   .7952
 1.0405                                   .7503
  .5499                                   .6771
 2.5232                                   .9339
 2.1061                                   .8452
 1.1514                                   .7293
 3.5298                                   .6784
  .7528                                   .6782
 2.0771                                   .7967
  .4917                                   .5779
 1.1888                                   .7049
  .7551                                   .5948
  .6132                                   .6291
  .5109                                   .6616
 2.9871                                   .7949
 1.1541                                   .6342
 1.5171                                   .7628
 5.5850                                   .6621
 2.0046                                   .7962
 1.6616                                   .7940
 2.9341                                   .8349
  AVE QUEUE LENGHT FOR ALL RUNS =       2.4085
  AVE UTILIZATION FOR ALL RUNS =         .7644

*FIN
```

Figure 4-5 (Continued).

GASP IV Simulation Program

The objective of this program is to simulate a Single Queue, Single Server system by using the GASP IV simulation language. The arrival of a message is exponentially distributed with mean time five minutes and the service time is exponentially distributed with mean time four minutes.

A Description of the GASP IV Simulation Program

The GASP IV Simulation Program is divided into the Main Program and four subroutines; they are: EVNTS, APR, BEGS, and FINS.

There are three files used for this program. File 1 is the event file, File 2 is for queueing the message, and File 3 is for service facility. The flow chart of this program is shown in Figure 4-6.

Main Program

The Main Program sets the card reader number (NCRDR) and the card printer number (NPRNT) and subroutine GASP is called.

Subroutine EVNTS

Subroutine EVNTS sends control to one of the three user written subroutines: ARR, BEGS, and FINS. The events of the simulation, in the order of their event code are:

20 - Arrival (ARR)

30 - Begin Service (BEGS)

40 - Finish Service (FINS)

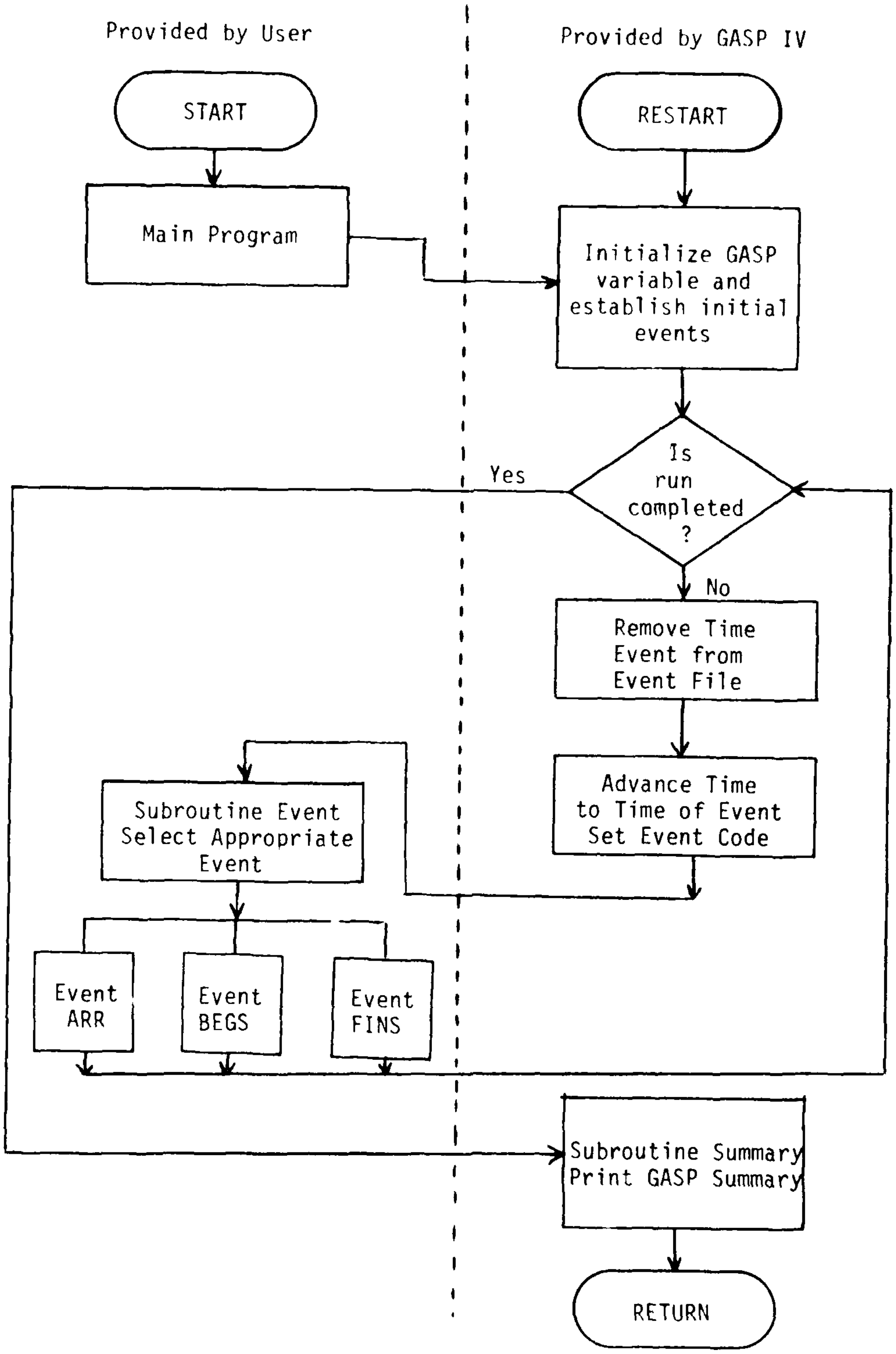

Provided by User
Provided by GASP IV
START
RESTART
Main Program
Initialize GASP variable and establish initial events
Is run completed ?
Yes
No
Remove Time Event from Event File
Advance Time to Time of Event Set Event Code
Subroutine Event Select Appropriate Event
Event ARR
Event BEGS
Event FINS
Subroutine Summary Print GASP Summary
RETURN

Figure 4-6 Flow Chart for GASP Single Queue, Single Server.

Subroutine ARR

The event arrival process of a message is accomplished in subroutine ARR. This subroutine records the arrival of a message and schedules the next arrival of a message. ARR also tests to see whether there are any messages in the Queue and if the Service Facility is free. If there are no messages in the Queue and the Service Facility is free, it schedules another arrival of a message; otherwise, ARR returns to subroutine BEGS.

Subroutine BEGS

The begin event is established in subroutine BEGS. This subroutine removes a message from the Queue, puts the message in the Service Facility, and schedules the finishing service.

Subroutine FINS

The finish event process is accomplished in subroutine FINS. This subroutine removes a message from the Service Facility and schedules begin service if there are any messages in the Queue.

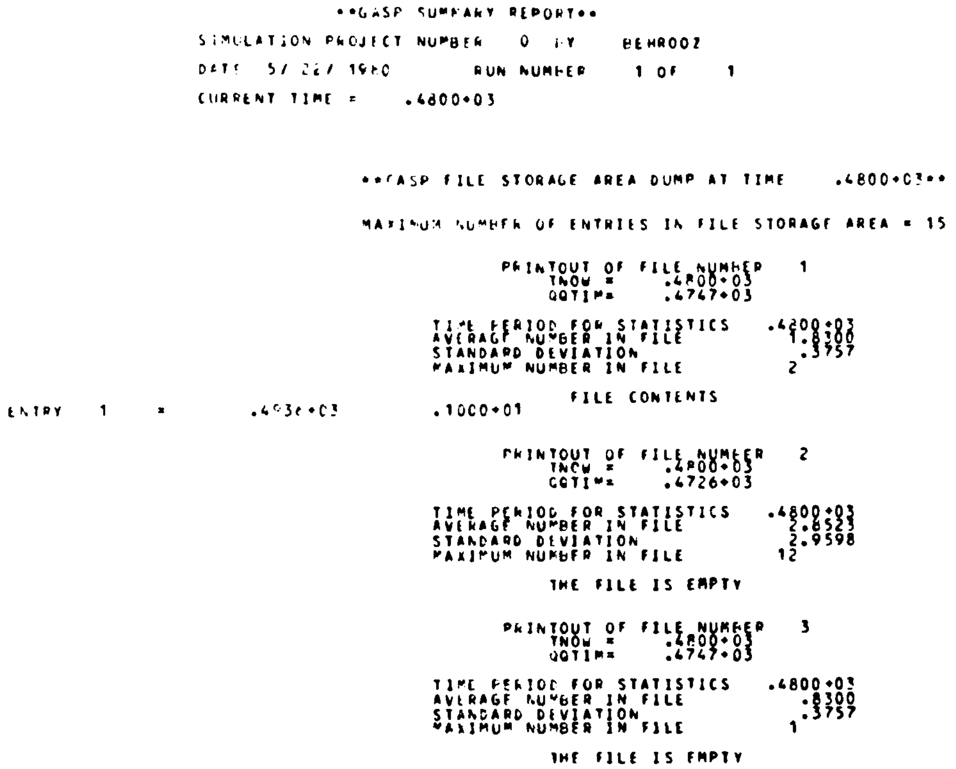
Simulation Report

Figure 4-7 presents the input data echo check, provided by subroutine DATIN and a printout of the files that are obtained at the end of subroutine DATIN.

Figure 4-8 is the GASP IV summary report. On the average, there are 1.83 events in file 1, with the standard deviation of .38 minutes. File 2 shows the average Queue length is 2.85 minutes, with the standard deviation of 2.95 minutes. The maximum number of messages in the Queue is 12. File 3 shows that the average utilization time is .83 minutes, with the standard deviation time of one minute. The maximum number of messages in the Service Facility is one.

```
                         SIMULATION PROJECT NUMBER   0  BY    BEHROOZ

                         DATE  5/ 22/ 1980             RUN NUMBER    1 OF     1
                         LLSUP=000000000000000     GASP IV VERSION 18MAY74

NNCLT=      0    NNSTA=      0     NNHIS=      0    NNPRM=      0    NNPLT=      0    NNSTR=      1    NNTRY=    15
NNATR=      2    NNFIL=      3     NNSET=    100    NNEQD=      0    NNEQS=      0    NFLAG=      0

KKRNK= (    1)      1       1

IINN = (    1)      1       4

MSTOP=      1    JJCLR=      1     JJBFG=      1    IICRD=      1    TTBEG=         .0000     TTFIN=        .4800+03
JJFIL=      1
IISED= 49763



                         **GASP FILE STORAGE AREA DUMP AT TIME      .0000    **

                         MAXIMUM NUMBER OF ENTRIES IN FILE STORAGE AREA =   1

                                      PRINTOUT OF FILE NUMBER     1
                                          TNOW =      .0000
                                          QQTIM=      .0000

ENTRY    1     =        .2000+01        .1000+01    FILE CONTENTS

                                      PRINTOUT OF FILE NUMBER     2
                                          TNOW =      .0000
                                          QQTIM=      .0000

                                          THE FILE IS EMPTY

                                      PRINTOUT OF FILE NUMBER     3
                                          TNOW =      .0000
                                          QQTIM=      .0000

                                          THE FILE IS EMPTY
```

Figure 4-7 Input Data Echo Check and Printout of Files at Time 0 for Single Bus Example.



```
                    **GASP SUMMARY REPORT**
              SIMULATION PROJECT NUMBER    0  BY    BEHROOZ
              DATE  5/ 22/ 1980        RUN NUMBER    1 OF    1
              CURRENT TIME =    .4800+03


                          **GASP FILE STORAGE AREA DUMP AT TIME    .4800+03**

                          MAXIMUM NUMBER OF ENTRIES IN FILE STORAGE AREA = 15

                                    PRINTOUT OF FILE NUMBER    1
                                          TNOW =     .4800+03
                                          QQTIM=     .4747+03

                                 TIME PERIOD FOR STATISTICS    .4800+03
                                 AVERAGE NUMBER IN FILE          1.8300
                                 STANDARD DEVIATION               .3757
                                 MAXIMUM NUMBER IN FILE         2

                                          FILE CONTENTS
ENTRY    1     =        .4936+03         .1000+01

                                    PRINTOUT OF FILE NUMBER    2
                                          TNOW =     .4800+03
                                          QQTIM=     .4726+03

                                 TIME PERIOD FOR STATISTICS    .4800+03
                                 AVERAGE NUMBER IN FILE          2.8523
                                 STANDARD DEVIATION              2.9598
                                 MAXIMUM NUMBER IN FILE        12

                                          THE FILE IS EMPTY

                                    PRINTOUT OF FILE NUMBER    3
                                          TNOW =     .4800+03
                                          QQTIM=     .4747+03

                                 TIME PERIOD FOR STATISTICS    .4800+03
                                 AVERAGE NUMBER IN FILE           .8300
                                 STANDARD DEVIATION               .3757
                                 MAXIMUM NUMBER IN FILE         1

                                          THE FILE IS EMPTY
```

Figure 4-8 GASP IV Summary Report for Single Bus Example.

GPSS II Simulation Program

The objective of this program is to simulate a single queue, single server system by using the GPSS II simulation language. The arrival of a message is exponentially distributed with mean time five minutes and the service time is exponentially distributed with mean time four minutes.

A Description of the GPSS II Simulation Program

The GPSS II Simulation Program performs the following functions:

(1) This program creates the arrival of a message by using the GENERATE block.

(2) GPSS II records the entries of messages in the QUEUE block.

(3) It begins service on the facility in the SEIZE block.

(4) The message uses the service facility in the ADVANCE block.

(5) The message frees the service facility in the RELEASE block.

(6) The simulation terminates when simulation time is over.

The block diagram for this program is given in Figure 4-9.

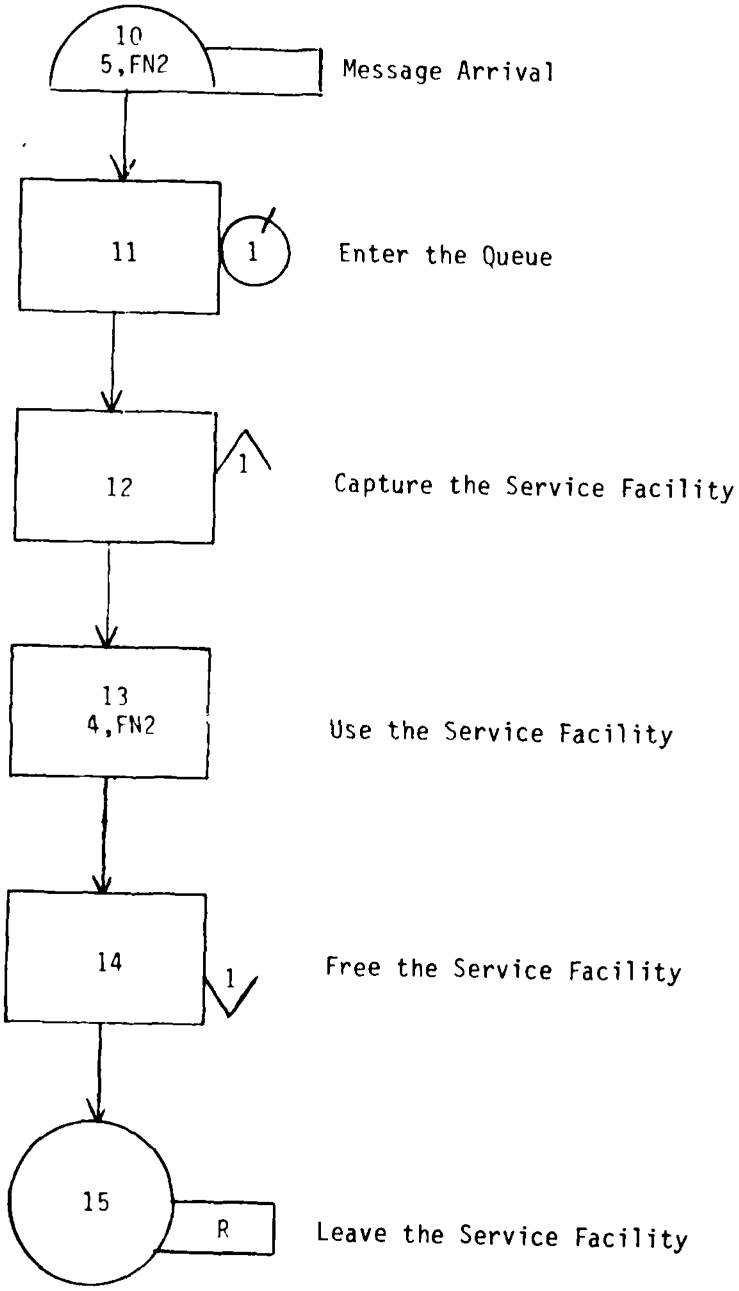


Figure 4-9 The Block Diagram for GPSS II Simulation Program.

Simulation Report

Figure 4-10 presents the GPSS II printout. The second and third lines of this printout are the transaction counts for all blocks. For example, at Block 2, 5 indicates the number of transactions currently at the block and 53 is the total number of transactions that entered the block.

Line six gives the following information: the facility number (1), the average utilization time (.76 minutes), the number of times the facility was used (48), and the average time for each transaction (3.37 minutes).

Line nine gives statistics for the Queue, measured by the block diagram. This includes the following:

(1) the number of the Queue used in the model (1)

(2) the largest number of messages in the Queue (6)

(3) the average number of messages in the Queue (1.45)

(4) the total number of messages entering in the Queue (53)

(5) the number of messages that have no waiting time (12)

(6) the percentage of messages that have no waiting time (22.64%)

(7) the average length of time that messages spent in the Queue (5.79 minutes)

(8) the average waiting time in the Queue (7.49 minutes), excluding the messages that do not have waiting time.

(9) there is no table (0)

(10) the current value of the Queue content (5)

```
TRANS COUNTS      BLOCK TRANS,TOTAL   BLOCK TRANS,TOTAL   BLOCK TRANS,TOTAL   BLOCK TRANS,TOTAL   BLOCK TRANS,TOTAL
                      1     0,   53       2     5,   53       3     0,   48       4     0,   48       5     0,   43
                      6     0,   48       7     0,    0       8     0,    0       9     0,    0      10     0,    0

FACILITY        AVERAGE          NUMBER        AVERAGE            TRANS      STRANS
      NR      UTILIZATION        ENTRIES      TIME/TRANS
       1         .7642              48           3.37               0           0

QUEUE      MAXIMUM     AVERAGE      TOTAL      ZERO      ZEROS        AVERAGE TIME/ENTRIES        TABLE      CURRENT
   NR      CONTENTS    CONTENTS     ENTRIES    ENTRIES   PERCENT     ALL ENT    NON ZERO ENT      NUMBER     CONTENTS
    1          6          1.45         53         12      22.64        5.79            7.49           0            5

FUTURE RANDOM NUMBER SEED IS (OCTAL)    163432722435
```

Figure 4-10 GPSS II Printout for Single Bus Example.

SIMSCRIPT II Simulation Program

The objective of this program is to simulate a Single Queue, Single Server system by using the SIMSCRIPT II simulation language. The arrival of a message is exponentially distributed with mean time five minutes and the service time is exponentially distributed with mean time four minutes.

A Description of the SIMSCRIPT II Simulation Program

The SIMSCRIPT II Simulation Program is divided into five parts; they are: Preamble, Main Program, Event Arrival, Event Departure, and Event Stop Simulation.

The Preamble defines every event, including Service Time, Queue Change, and Sum of Queue, as variables. It also defines the status of the model and all integer variables.

The Main Program schedules the arrival of a message, the desired number of hours for the simulation run, and the start of the simulation.

The Event Arrival schedules the arrival of a message and creates a message. It files the message in the Queue and records the total number of entries. The Event Arrival computes the sum of the Queue length and schedules the departure of the message. Figure 4-11 shows the flow chart for Event Arrival.

The Event Departure lets the status be idle if the Queue is empty; otherwise, it removes the first message from the Queue and updates the last Queue change. The Event Departure destroys the message and determines the service time. It also schedules the

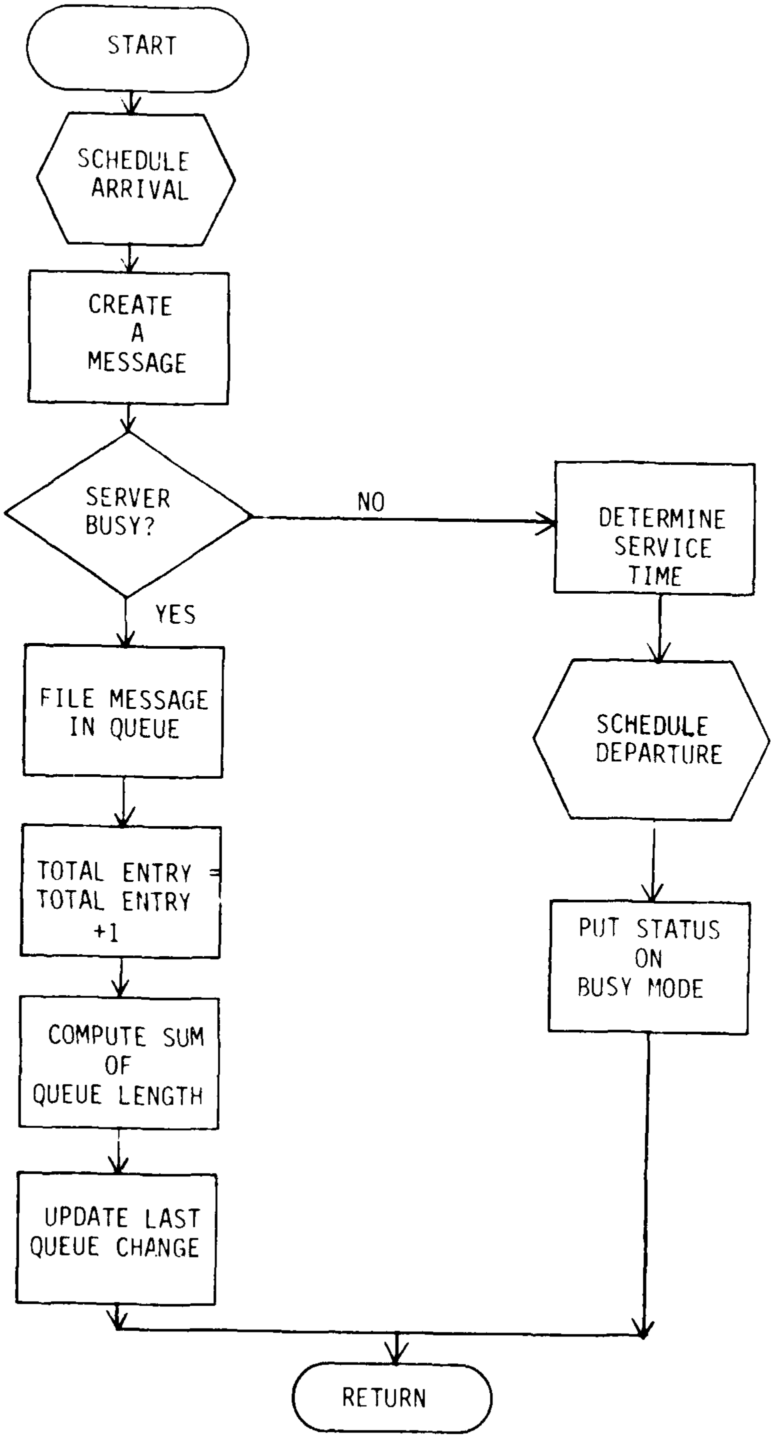


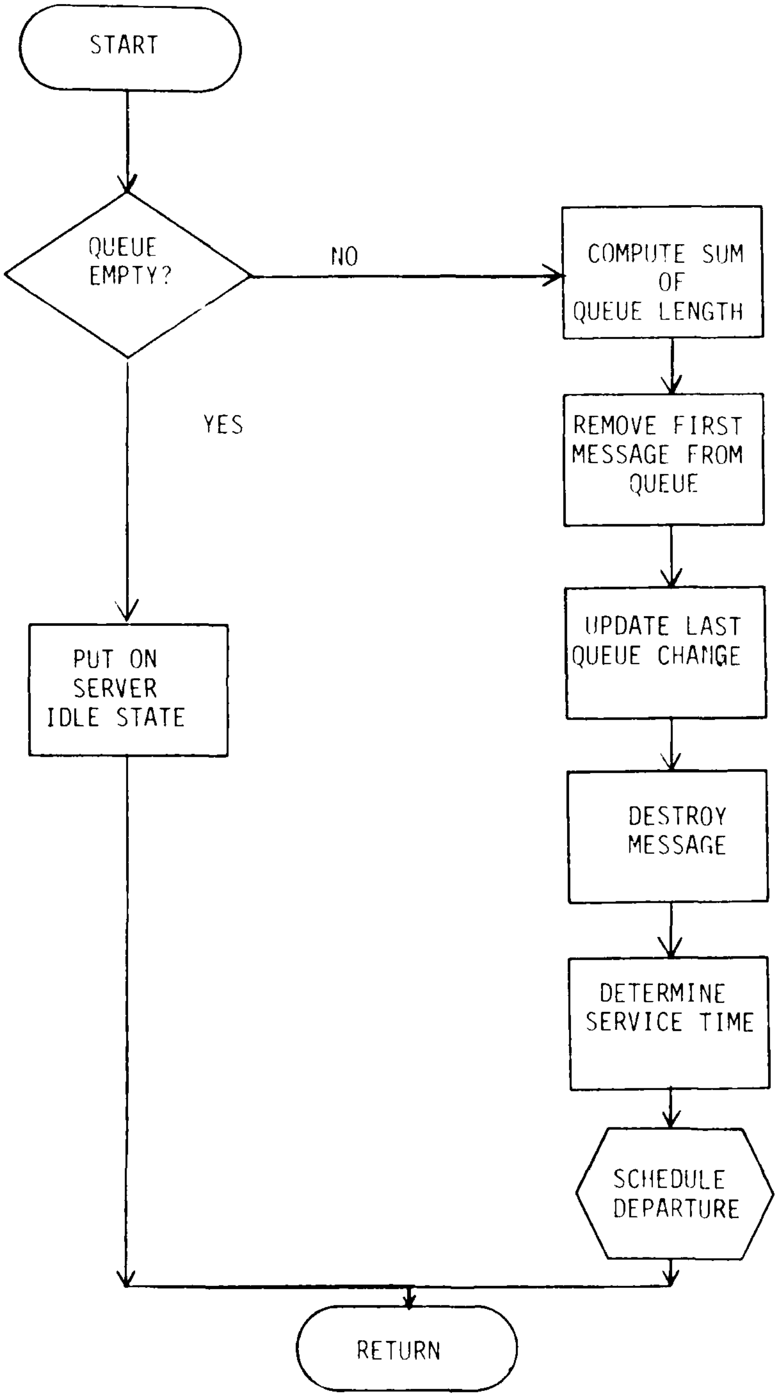
Figure 4-11 Flow Chart for Event Arrival for Single Bus Example.

departure of the message. Figure 4-12 shows the flow chart of Event Departure.

The Event Stop Simulation gives the simulation statistics, such as, the average Queue length, utilization time, the maximum number of messages in the Queue, and the total entries.

Simulation Report

The average Queue length for this simulation program is 3.69 minutes and the average utilization time is 0.94 minutes. The total number of messages entered is 99 and the maximum number in the Queue is 9.



Fiqure 4-12 Flow Chart for Event Departure for Single Server Example.

CHAPTER V

SUMMARY AND CONCLUSION

In the preceding chapters of this report five simulation languages are presented: GASP IV, GPSS II, SIMSCRIPT II, ADA, and ECSS II. Additionally, a simulation of a simple bus, single queue system is shown utilizing FORTRAN IV, GASP IV, GPSS II, and SIMSCRIPT II. The objective of this simulation is to obtain information on queue length and bus utilization and compare the various programming languages. The ADA and ECSS II languages are not available on the Mississippi State University Univac 1108 System, thus there are no simulation runs in these languages.

The primary thrust in this research effort has been the utilization of AFAL's MUXSIM simulation program. MUXSIM was copied from the AFAL DEC System 10 onto magnetic tape and transported to Mississippi State University. Considerable time and effort was expended in adapting MUXSIM to the UNIVAC 1108 system. Due to the non-availability of interactive terminals at Mississippi State University, MUXSIM runs were made in batch mode using card decks.

The following results were achieved with MUXSIM operating on the UNIVAC 1108 system:

1. The dynamic portions of MUXSIM, MUXDA and MUXDB, were software modified for use with the UNIVAC 1108 based GASP IV. Simulation runs of MUXDA and MUXDB are listed in Chapter III of this report.

2. GASP IV user subroutines are written in FORTRAN language. Additional FORTRAN subroutines for expanded plots of MUXDA bus statistics collected by GASP IV are listed (p. 154). See the Appendix for all program listings.
3. Data files for MUXDA and MUXDB were linked to GASP IV.

Finally, the ADA and ECSS II languages were considered as possible simulation tools for future avionics multiplex data bus studies. Both ADA and ECSS II are general purpose languages with attributes which make them candidates for consideration. ADA has programming features similar to COBOL and PASCAL. The ECSS II language relies on a computer system with a SIMSCRIPT compiler. From a practical point of view, it appears ECSS II offers little advantage over SIMSCRIPT. As mentioned previously, no simulation runs were made with ADA and ECSS II; the discussion of these languages is based on information obtained from references [8] [9].

The foregoing statements summarize the work accomplished under this grant and cover the work statements outlined in the proposal. An additional study was made for comparative purposes utilizing GASP IV, GPSS II, FORTRAN, and SIMSCRIPT. The results of this comparison are shown in Chapter IV.

APPENDIX

LIST OF PROGRAMS AND SUBROUTINES

MUXDA

```
IT      PROC
        DIMENSION NSET(1)
        COMMON QSET(5000)
        EQUIVALENCE (NSET(1),QSET(1))
        COMMON/GCOM1/ATRIB(25),JEVNT,MFA,MFE(100),MLE(100),MSTOP,NCRDR,
       $NNAPO,NNAPT,NNATR,NNFIL,NNQ(100),NNTRY,NPRNT,PPARM(50,4),TNOW,
       $TTBEG,TTCLR,TTFIN,TTRIB(25),TTSET
        COMMON /GCOM2/ DD(100),DDL(100),DTFUL,DTNOW,ISEES,LFLAG(50),NFLAG,
       $NNEQD,NNEQS,NNEQT,SS(100),SSL(100),TTNEX
        COMMON/GCOM3/ AAERR,DTMAX,DTMIN,DTSAV,IITES,LLERR,LLSAV,LLSEV,RRE
       $RR,TTLAS,TTSAV
        COMMON/GCOM4/ IIPLT(10),HHLOW(25),HHWID(25),IICRD,IITAP(10),JJCEL
       $(500),LLABC(25,2),LLABH(25,2),LLABP(11,2),LLAPT(25,2),LLPHI(10),LL
       $PLO(10),LLPLT,LLSUP(15),LLSYM(10),MMPTS,NNCEL(25),NNCLT,NNHIS,NNPL
       &T,NNPTS(10),NNSTA,NNVAR(10),PPHI(10),PPLO(10)
        COMMON/GCOM5/ IIEVT,IISED(6),JJBEG,JJCLR,MMNIT,MMON,NNAME(3),NNCF
       &I,NNDAY,NNPT,NNSET,NNPRJ,NNPRM,NNRNS,NNRUN,NNSTR,NNYR,SSEED(6)
        COMMON/GCOM6/EENQ(100),IINN(100),KKRNK(100),MMAXQ(100),QQTIM(100)
       &,SSOBV(25,5),SSTPV(25,6),VVNQ(100)
 END
```

```
C       PROGRAM MUXDA
        INCLUDE IT
C
C
C-----------------------INPUT CARD DEFINITION--------------------------
C       CARD TYPE ARE DEFINED BY RELATIVE POSITION WITHIN THE DICTIONARY  .
C       DECK.  THESE CARDS ARE LOCATED AFTER THE GASP IV CONTROL CARDS FOR
C       SIMULATION RUN 1.
C
C
C-----CARD TYPE I DEFINES THE FUI TIME LENGTH AND DEMAND MESSAGE
C       TRANSFER MODE (1,2).  (F10.0,8X,I2) (ONE CARD)
C       IF MODE 1 IS SELECTED TWO RUNS WILL ENSUE. (ONE MODE 1, THE SECOND
C       MODE 2) IF MODE 2 IS SELECTED ONE RUN WILL ENSUE. (MODE 2)
C                 MODE 1= FIRST-IN-FIRST-OUT.
C                 MODE 2= LONGEST MESSAGE THAT CAN BE TRANSMITTED IN THE
C                         IN THE REMAINING TIME GOES FIRST.
C
C
C
C-----CARD TYPE II DEFINES THE DURATION OF THE FIXED MESSAGE SEQUENCE
C       FOR EACH FUI. (THIS REQUIRES A CARD FOR EACH FUI AND THE CARDS
C       MUST BE SEQUENCED BY ASCENDING FUI NUMBER.) (F10.0)
C                 0.0 -ENTRY INDICATES THE END OF CARD TYPE II SEQUENCE.
C
C
C-----CARD TYPE III DEFINES THE DEMAND MESSAGES. (4F10.0) (EACH CARD
C       DEFINES A DEMAND MESSAGE.)
C       THE ENTRIES IN EACH CARD ARE DEFINED AS:
C                 FIELD 1= MEAN ARRIVAL TIME. (UNIFORMLY DISTRIBUTED)
C                 FIELD 2= MAXIMUN DEVIATION FROM MEAN ARRIVAL (+,-).
C                 FIELD 3= DEMAND MESSAGE LENGTH.
C                 FIELD 4= (UNDEFINED FOR THIS MODEL.)
C
C
C
C                 0.0 - ENTRY IN FIELD 1 INDICATES THE END OF CARD TYPE
C                       III SEQUENCE.
C
C
        NCRDR=5
        NPRNT=6
        CALL GASP
        CALL EXIT
        END
```

```
       SUBROUTINE EVNTS (IX)
       I=IX/100
       J=IX-(I*100)
       IF(I.EQ.0) CALL ERROR(1)
       GO TO (100,200,300,400),I
100    CALL FUIT(J)
       GO TO 99
200    CALL FUIFRE (J)
       GO TO 99
300    CALL ENDDM (J)
       GO TO 99
400    CALL DMARIV (J)
99     RETURN
       END



       SUBROUTINE ENDDM(NM)
C
C------A 300 TYPE EVENT (END A DEMAND MESSAGE)
C      PERFORMS THE END OF DEMAND MESSAGE ARRIVAL PROCESSING.
C
       INCLUDE IT
       COMMON /MUX1/FUI1T,NDM,FUI,NFUIS,FUINXT,NFUI,MODE
       COMMON /MUX2/DM(5,20),FUIFX(25)
       DELAY=TNOW-ATRIB(1)
       I=NM+10
       CALL HISTO(DELAY,I)
C      FREE TIME STARTS AFTER FINISH OF A DEMAND MESSAGE
       ATRIB(1)=TNOW
       ATRIB(2)=200.+FLOAT(NFUI)
       CALL FILEM(1)
       RETURN
       END

       SUBROUTINE FUIFRE(NF)
C
C------A 200 TYPE EVENT (START OF FREE TIME IN THIS FUI)
C      PROCESSES THE END OF A MESSAGE TRANSMISSION AND SCHEDULES THE
C      NEXT DEMAND MESSAGE TRANSMISSION.
C
       INCLUDE IT
       COMMON /MUX1/FUI1T,NDM,FUI,NFUIS,FUINXT,NFUI,MODE
       COMMON/MUX2/DM(5,20),FUIFX(25)
       COMMON /MUX3/ DMSENT(20),FUIWAT(20)
C------X IS THE REMAINING FREE FUI TIME
       X=FUINXT-TNOW
C------EXIT IF NOTHING QUED OR NO TIME LEFT
       IF(MFE(2).LE.0)GO TO 2
       IF(X.LE.0.)GO TO 2
C      SELECT ENTRY FROM QUE
       GO TO (11,12),MODE
11     CONTINUE
```

```
      SUBROUTINE INTLC
C
C
C     READS IN SIMULATION DATA CARDS AND SETS UP INITIAL CONDITIONS FOR
C     SIMULATION EITHER FROM THE INPUT DATA CARDS OR BY ALGEBRAIC
C     STATEMENTS.
C
C
      INCLUDE IT
      COMMON /MUX1/FUIIT,NDM,FUI,NFUIS,FUINXT,NFUI,MODE
      COMMON /MUX2/DM(5,20),FUIFX(25)
      COMMON /MUX3/ DMSENT(20),FUIWAT(20)
      IRD=NCRDR
      IF(NNRUN.NE.1)GO TO 20
      NFUIS=0
C
C     READS IN CARD TYPE I WHICH DEFINES THE FUI DURATION AND RUN MODE.
C
      READ(IRD,403)FUI,MODE
403   FORMAT(F10.5,5X,I5)
      IF(MODE.LE.0)MODE=1
      WRITE(NPRNT,300)FUI,MODE
300   FORMAT('0 FUI=',F10.6,' MODE OF',I3,/,
     X '0 FUI NO.           FREE FUI START')
C
C     READS IN CARD TYPE II WHICH DEFINES FUI FIXED MESSAGE SEQUENCE
C     DURATION AND QUANTITY OF FUIS.
C
1     READ(IRD,100) X
100   FORMAT(F10.5)
      IF(X.LE.0.0) GO TO 2
      NFUIS=NFUIS+1
      WRITE(NPRNT,301)NFUIS,X
301   FORMAT(I3,F20.6)
      FUIFX(NFUIS)=X
      IF(X.GT.FUI) CALL ERROR(2)
      GO TO 1
2     CONTINUE
      FUIIT=TNOW
      DO 3 I=1,NFUIS
      X=FUIIT+FUI*FLOAT(I-1)
3     CALL NXTFUI(I,X)
4     NDM=1
C
C     READS IN CARD TYPE III WHICH DEFINES THE DEMAND MESSAGES.
C
5     READ(IRD,200)(DM(J,NDM),J=1,4)
200   FORMAT(4F10.0)
      IF(DM(1,NDM).LE.0.0) GO TO 10
      NDM=NDM+1
      GO TO 5
10    CONTINUE
      NDM=NDM-1
      WRITE(NPRNT,302)    NDM
302   FORMAT('1DM NO.',10X,'4 PARAMETERS OF A DM      - NDM=',I4)
      DO 11 I=1,NDM
      WRITE(NPRNT,303)I,(DM(J,I),J=1,4)
303   FORMAT(I5,4(5X,F10.6))
11    CALL NXTDM(I)
30    CONTINUE
      DO 15 I=1,20
      DMSENT(I)=0.
15    FUIWAT(I)=0.
      RETURN
C
20    CONTINUE
      NNPTS(1)=NNPTS(1) - 1
      MODE=MODE+1
      FUIIT=TNOW
      DO 23 I=1,NFUIS
      X=FUIIT+FUI*FLOAT(I-1)
23    CALL NXTFUI(I,X)
      DO 24 I=1,NDM
24    CALL NXTDM(I)
      GO TO 30
      END
      I=MFE(2)
      GO TO 20
12    CONTINUE
      I=NFIND(X,3,2,1,0,0)
20    CONTINUE
C------COPY SELECTION INTO ATRIBUTES AREA
      IF(I.EQ.0)GO TO 2
      CALL COPY(I)
C------TEST IF WILL FIT
      IF(ATRIB(1).GE.X)GO TO 2
      CALL RMOVE(I,2)
C------DML IS DEMAND MESSAGE LENGTH
      DML=ATRIB(1)
      ATRIB(1)=TNOW+DML
      ATRIB(2)=300.+ATRIB(2)
      CALL FILEM(1)
      DMSENT(NF)=DMSENT(NF)+DML
      RETURN
2     CONTINUE
      FUIWAT(NF)=FUIWAT(NF)+X
      RETURN
      END


      SUBROUTINE FUIT(NF)
C
C------A 100 TYPE EVENT (START OF THIS FUI)
C      PERFORMS THE PROCESSING INVOLVED WITH FUI STARTUP.
C
      REAL XX(4)
      INCLUDE IT
      COMMON /MUX1/FUI1T,NDM,FUI,NFUIS,FUINXT,NFUI,MODE
      COMMON /MUX2/DM(5,20),FUIFX(25)
      COMMON /MUX3/ DMSENT(20),FUIWAT(20)
      X=FUI*FLOAT(NFUIS)
C------FOLLOWING STATEMENT IS SAVING FUI 1 TIME TO PREVENT DRIFT OF
C------FUI TIMES WITH RESPECT TO EACH OTHER DUE TO ROUNDOFF
      IF(NF.EQ.1) FUI1T=TNOW+X
C------TEST FOR FIRST CYCLE THROUGH MAJOR CYCLE
C------SKIP PLOT CALL IF FIRST MAJOR CYCLE
      IF(FUI1T.LE.(TTBEG+X)) GO TO 1
C------COLECT ONLY 100 SETS OF POINTS
      IF(NNPTS(1).GE.100.)GO TO 1
C------X IS THE FREE FUI TIME AVALIABLE FOR SCHEDULE OF DEMAND MESSAGES
      X=FUI-FUIFX(NF)
C------PLOT ARGUMENTS
C------1 IS QUE NUMBER AT FUI TIME
C------2 IS THE DEMAND MESSAGE LENGTHS SENT PERCENTAGE OF AVALABLE
C------3 IS UNUSED TIME AS PERCENTAGE OF NON FIXED TIME
      XX(1)=FLOAT(NNQ(2))
      XX(2)=DMSENT(NF)/X
      XY(3)=FUIWAT(NF)/X
      CALL GPLOT(XX,TNOW,1)
1     CONTINUE
      CALL HISTO(DMSENT(NF),NF)
C------ZERO TOTAL ACCUMULATORS
      DMSENT(NF)=0.0
      FUIWAT(NF)=0.0
C------X IS NOW TIME OF NEXT FUI FOR THIS FUI NUMBER
      X=FUI1T+FUI*FLOAT(NF-1)
      CALL NXTFUI(NF,X)
C------SCHEDULE THE FREE FUI TIME START FOR THIS FUI
      ATRIB(1)=TNOW+FUIFX(NF)
      ATRIB(2)=200.+FLOAT(NF)
      ATRIB(3)=FUI-FUIFX(NF)
      CALL FILEM(1)
C------FUINXT IS WHEN NEXT FUI STARTS
      FUINXT=TNOW+FUI
C------NFUI IS CURRENT FUI INTERVAL NUMBER
      NFUI=NF
      RETURN
      END
      SUBROUTINE DMARIV(ND)
C------A 400 TYPE EVENT (ARRIVAL OF A DEMAND MESSAGE)
C      PROCESSES THE ARRIVAL OF DEMAND MESSAGES AND CALLS FOR SCHEDULING
C      OF THE NEXT DEMAND MESSAGE ARRIVAL.
C
C   DM IS THE DEMAND MESSAGE DEFINITION
C   DM(X,Y) IS DEFINED AS:
C      Y=DM=
C      X=1=ARRIVAL TIME MEAN
C      X=2=DISTRIBUTION CODE VALUE ABOUT MEAN
C      X=3=MESSAGE LENGTH
C      X=4=
C
      INCLUDE IT
      COMMON /MUX1/FUITT,NDM,FUI,NFUIS,FUINXT,NFUI,MODE
      COMMON /MUX2/DM(5,20),FUIFX(25)
C------SCHEDULE THE NEXT DEMAND MESSAGE ARRIVAL
      CALL NXTDM(ND)
C------PUT CURRENT D.M. ON WAITING QUE
      ATRIB(1)=DM(3,ND)
      ATRIB(2)=FLOAT(ND)
      ATRIB(3)=TNOW
      CALL FILEM(2)
      RETURN
      END

      SUBROUTINE NXTFUI(I,WHEN)
C
C      SCHEDULES THE START OF THE NEXT FUI.
C
      COMMON /MUX1/FUITT,NDM,FUI,NFUIS,FUINXT,NFUI,MODE
      COMMON /MUX2/DM(5,20),FUIFX(25)
      INCLUDE IT
1     ATRIB(1)=WHEN
      ATRIB(2)=100.+FLOAT(I)
      CALL FILEM(1)
      RETURN
      END

      SUBROUTINE NXTDM(I)
C
C      SCHEDULES ARRIVAL OF THE NEXT DEMAND MESSAGE.
C
      COMMON /MUX1/FUITT NDM,FUI,NFUIS,FUINXT,NFUI,MODE
      COMMON /MUX2/DM(5,20),FUIFX(25)
      INCLUDE IT
      PHI=DM(2,I)
      PLOW=-PHI
      J=MOD(I,5)+1
      ATRIB(1)=TNOW+DM(1,I)+UNFRM(PLOW,PHI,J)
      ATRIB(2)=400.+FLOAT(I)
      CALL FILEM(1)
      RETURN
      END
```

MUXDB

```
1 [illegible] (-X.)        KI
:Y     PROC
       DIMENSION NSET(1)
       COMMON QSET(5000)
       EQUIVALENCE (NSET(1),QSET(1))
       COMMON/GCOM1/ATRIB(25),JEVNT,MFA,MFE(100),MLE(100),MSTOP,NCRDR,
      $NNAPO,NNAPT,NNATR,NNFIL,NNQ(100),NNTRY,NPRNT,PPARM(50,4),TNOW,
      $TTBEG,TTCLR,TTFIN,TTRIB(25),TTSET
       COMMON /GCOM2/ DD(100),DDL(100),DTFUL,DTNOW,ISEES,LFLAG(50),NFLAG,
      $NNEQD,NNEQS,NNEQT,SS(100),SSL(100),TTNEX
       COMMON/GCOM3/ AAERR,DTMAX,DTMIN,DTSAV,IITES,LLERR,LLSAV,LLSEV,RRE
      $RR,TTLAS,TTSAV
       COMMON/GCOM4/ DTPLT(10),HHLOW(25),HHWID(25),IICRD,IITAP(10),JJCEL
      $(500),LLABC(25,2),LLABH(25,2),LLABP(11,2),LLAPT(25,2),LLPHI(10),LL
      $PLO(10),LLPLT,LLSUP(15),LLSYM(10),MMPTS,NNCEL(25),NNCLT,NNHIS,NNPL
      $T,NNPTS(10),NNSTA,NNVAR(10),PPHI(10),PPLO(10)
       COMMON/GCOM5/ IIEVT,IISED(6),JJBEG,JJCLR,MMNIT,MMON,NNAME(3),NNCF
      $I,NNDAY,NNPT,NNSET,NNPRJ,NNPRM,NNRNS,NNRUN,NNSTR,NNYR,SSEED(6)
       COMMON/GCOM6/EENQ(100),IINN(100),KKRNK(100),MMAXQ(100),QQTIM(100)
      $,SSOBV(25,5),SSTPV(25,6),VVNQ(100)
END


C       PROGRAM MUXDB
        INCLUDE IT
        COMMON /MUX1/NDM,DM(4,25),TMSENT(25)
        COMMON /MUX2/FUI1T,FUI,NFUIS,NFIX(25),FIX(25,25),ICURF
        COMMON /MUX3/NNOISE,NSBUS(15),SMEAN(15),SMVAR(15),SLNG(15),SLVAR
       1(15)
        COMMON /MUX4/NTM,TMEAN(15),TMVAR(15),TLEN(15),TLVAR(15)
        COMMON /MUX6/ILBUS,IRBUS
        COMMON /MUX5/NTR,TRES(15),TRAR(15),NUMV(15),IOK(15)
        COMMON /MUX7/IBUSY,MSGTYP,MSGNOW
        DATA ISW/1/
C-----------------------INPUT CARD DEFINITION-----------------------
C
C       INPUT CARD TYPE IS DEFINED BY THE LETTER ENTERED IN COLUMN 1
C       DEFINED AS FOLLOWS:
C
C
C
C       X = EXIT,FINISHED WITH INPUT.
C
C       D = DEMAND MESSAGE    F(T11,4F10.0) FOR  MEAN,VARIANCE,LENGTH,TERM.NO.
C
C       F = FIXED MESSAGE     F(T6,I5,14F5.0) FOR FUI NUMBER AND 14 MESSAGE
C                                             LENGTHS.
C
C       T = TERMINAL DOWN     F(T6,I5,4F10.0) FOR TERMINAL NUMBER, MEAN AND
C                                             VARIANCE OF OCCURANCE, AND
C                                             LENGTH AND VARIANCE OF DOWN.
C
C       N = NOISE             F(T6,I5,4F10.0) FOR BUS INDICATOR(1,2,3=LEFT,
C                                             RIGHT,BOTH), MEAN AND
C                                             VARIANCE OF OCCURANCE, AND
C                                             LENGTH AND VARIANCE OF
C                                             DURATION.
C
C       U = NUMBER OF FUIS    F(T6,I5,F10.0)  FOR NUMBER OF FUIS, FUI
C                                             INTERVAL (FUNDIMENTAL UPDATE
C                                             INTERVALS).
C
C       R = RESPONSE TIME     F(T6,I5,2F10.0) FOR TERMINAL NUMBER, MEAN
C                                             RESPONSE TIME, VARIANCE OF
C                                             RESPONSE TIME.
C       C = COMMENT CARD IN DECK - NO ACTION
C
C
C---------------------EVENT DEFINITIONS------------------------------
C
C       EVENT TYPES ARE DEFINED BY HUNDREDS (100) AND EVENT SUB-TYPES
C       ARE DEFINED BY UNITS (1), AS FOLLOWS:
C
C                      100 -  WATCH DOG TIMER
C                             1.TIME
C                             2.EVENT NUMBER
C                             3.MESSAGE NUMBER
C
C                      200 -  CALL TERMINAL
C                             1.TIME
C                             2.EVENT NUMBER + TERMINAL DESTINATION
C                             3.MESSAGE NUMBER+1000'S OF HITS LEFT + 1000000'S
C                               OF HITS RIGHT.
C
C                      300 -  TERMINAL RESPONSE
C                             1.TIME
C                             2.EVENT NUMBER + TERMINAL RESPONDING
C                             3.MESSAGE NUMBER +1000'S OF HITS LEFT + 1000000'S
C                               OF HITS RIGHT
C
```

```
C
C                 400 -   NOISE START
C                          1.TIME
C                          2.EVENT NUMBER + BUS DESIGNATION (1-3)
C                          3.NOISE NUMBER
C
C                 500 -   NOISE END
C                          1.TIME
C                          2.EVENT NUMBER + BUS DESIGNATION (1-3)
C                          3.NOISE NUMBER
C                 600 -   TERMINAL DOWN
C                          1.TIME
C
C                 700 -   TERMINAL UP
C                          1.TIME
C                          2.EVENT NUMBER + TERMINAL NUMBER
C
C                 800 -   FUI TIME START
C                          1.TIME
C                          2.EVENT NUMBER + FUI NUMBER
C
C                1000 -   DEMAND MESSAGE ARRIVAL
C                          1.TIME
C                          2.EVENT + DEMAND MESSAGE NUMBER
C
C                 MESSAGE NUMBERS-  100'S FOR FIXED TYPE
C                                   200'S FOR DEMAND TYPE
C
      NCRDR=5
      NPRNT=6
      CALL GASP
      CALL EXIT
      END

      SUBROUTINE INTLC
C
C
C     READS IN THE SIMULATION DATA CARDS AND SETS UP THE INITIAL
C     CONDITIONS FOR THE SIMULATION EITHER FROM THE SIMULATION INPUT
C     DATA CARDS OR BY ALGEBRAIC STATEMENTS
C
C
C
      INTEGER CARD(80),SCRTCH(80)
      INCLUDE IT
      COMMON /MUX1/NDM,DM(4,25),TMSENT(25)
      COMMON /MUX2/FUI1T,FUI,NFUIS,NFIX(25),FIX(  ,25),ICURF
      COMMON /MUX3/NNOISE,NSBUS(15),SMEAN(15),SMVAR(15),SLNG(15),SLVAR
     1(15)
      COMMON /MUX4/NTM,TMEAN(15),TMVAR(15),TLEN(15),TLVAR(15)
      COMMON /MUX5/NTR,TRES(15),TRVAR(15),NUMV(15),IOK(15)
      COMMON /MUX6/ILBUS,IRBUS
      COMMON /MUX7/IBUSY,MSGTYP,MSGNOW
      COMMON /MUXUSE/FUIPRS,FME,DME
      COMMON /MUXRES/IRESE
      COMMON /STAT/NMSGH,NNL,NNR,NNP,NTO,NMSG
      DATA NC/1HC/
      DATA NX,ND,NT,NF,NR,NN,NU/1HX,1HD,1HT,1HF,1HR,1HN,1HU/
      NNM=0
      NNLM=0
      NNRM=0
      NCM=0
      FUI1T=TNOW
      NNOISE=0
      NFUIS=0
      NTM=0
      NTR=0
      IEBUS=0
      IFBUS=0
      IBUSY=0
      MSGTYP=1
      FUIPRS=TNO.
      IRESE=0
      NMSGH=0
      NML=0
      NNR=0
      NNB=0
      NTU=0
      NMSG=0
      DO 2 I=1,25
      FIX(1,I)=0.0
2     NFIX(I)=0
      DO 3 I=1,15
3     ION(I)=0
      WRITE(NPRNT,500)
500   FORMAT('1',T20,'INPUT CARDS')
      IPAG=0
1     CONTINUE
      READ(NCRDR,100,END=99)CARD
100   FORMAT(80A1)
      ENCODE(80,100,SCRTCH)CARD
      IC=CARD(1)
      WRITE(NPRNT,501)CARD
 501  FORMAT(1X,T11,80A1)
      IPAG=IPAG+1
      IF(IPAG.LE.52)GO TO 502
      WRITE(NPRNT,500)
      IPAG=0
502   CONTINUE
      IF(IC.EQ.NX)GO TO 99   .
      IF(IC.EQ.NC)GO TO 1
      IF(IC.EQ.ND) GO TO 10
      IF(IC.EQ.NT) GO TO 30
      IF(IC.EQ.NF) GO TO 20
      IF(IC.EQ.NR) GO TO 40
      IF(IC.EQ.NN) GO TO 50
      IF(IC.EQ.NU) GO TO 60
      CALL ERROR(10)
      GO TO 1
C
C                    10 = DEMAND MESSAGE CARD
C
10    CONTINUE
      NDM=NDM + 1
      DFCODE(80,101,SCRTCH)(DM(I,NDM),I=1,4)
101   FORMAT(T11,4F10.0)
      GO TO 1
C
C                    20 = FIXED MESSAGE CARD
C
20    CONTINUE
      DFCODE(80,102,SCRTCH)I
102   FORMAT(T6,I5)
      J=NFIX(I)+1
      K=J+13
      DECODE(80,103,SCRTCH)(FIX(L,I),L=J,K)
103   FORMAT(T11,14F5.0)
      NFIX(I)=K
      GO TO 1
C
C                    30 = TERMINAL DOWN CARD
C
30    CONTINUE
      NTM=NTM+1
      DECODE(80,104,SCRTCH)I          ,TMEAN(NTM),TMVAR(NTM),TLEN(NTM),
     1TLVAR(NTM)
104   FORMAT(T6,I5,4F10.0)
      GO TO 1
C
C                    40 = TERMINAL RESPONSE TIME
C
40    CONTINUE
      NTR=NTR+1
      DECODE(80,105,SCRTCH)I          ,TRES(NTR),TRVAR(NTR)
105   FORMAT(T6,I5,2F10.0)
      GO TO 1
```

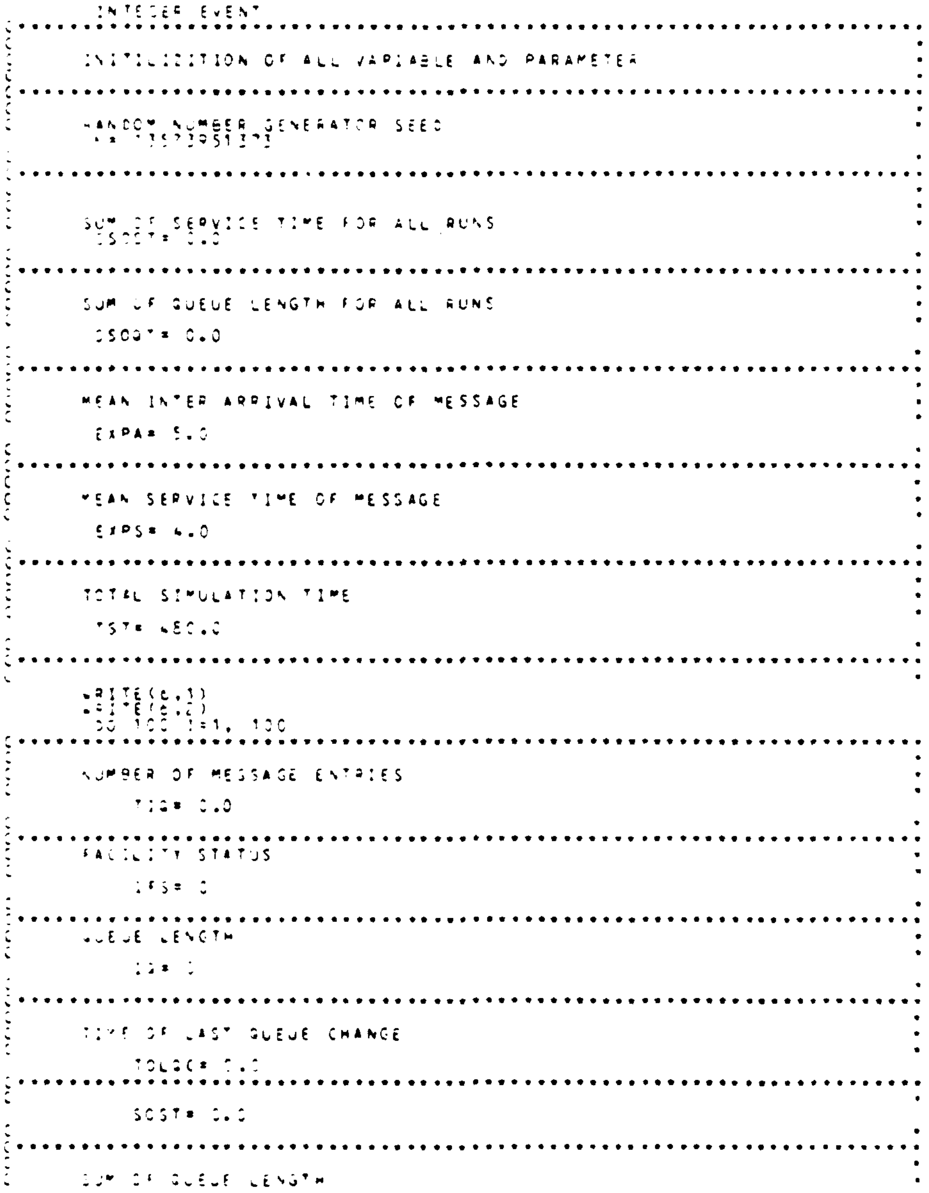
```
C
C                    50 = NOISE CARD
C
50    CONTINUE
      NNOISE=NNOISE+1
      DECODE(80,106,SCRTCH)NSBUS(NNOISE),SMEAN(NNOISE),SMVAR(NNOISE),
     1 SLNG(NNOISE),SLVAR(NNOISE)
106   FORMAT(T6,I5,4F10.0)
      GO TO 1
C
C                    60 = FUI NUMBER CARD
C
60    CONTINUE
      DECODE(80,107,SCRTCH)NFUIS,FUI
107   FORMAT(T6,I5,F10.0)
      GO TO 1
C
C                    99 = END INPUT
C
C
C                    DEMAND MESSAGES
C
99    CONTINUE
      IF(NDM.LE.0) GO TO 200
      WRITE(NPRNT,105)
105   FORMAT('1',T30,'DEMAND MESSAGES',/,'0 MEAN',T20,'VARIANCE',T40,
     1'LENGTH',T60,'TERMINAL NO.')
      DO 201 I=1,NDM
      WRITE(NPRNT,109)(DM(J,I),J=1,4)
109   FORMAT(1X,F10.4,T20,F10.4,T40,F10.4,T60,F10.4)
201   CONTINUE
C
C                    FIXED MESSAGES
C
200   CONTINUE
      NFUI=NFUIS
      IF(NFUI.LE.0) CALL ERROR(10)
      WRITE(NPRNT,111)
111   FORMAT('1 FUI - NO. FIXED MESSAGES - MESSAGE LENGTHS',/,'0')
      DO 221 I=1,NFUI
      J=NFIX(I)
      DO 224 K=1,J
      L=J-K+1
      IF(FIX(L,I).GT.0.)GO TO 225
224   CONTINUE
      L=0
225   NFIX(I)=L
      J=L
      IF(J.LE.0) GO TO 222
      WRITE(NPRNT,110)I,J,(FIX(L,I),L=1,J)
110   FORMAT(1X,I3,T5,I3,4(1X,T10,F10.4,T30,F10.4,T50,F10.4,T70,F10.4,/)
     1)
      GO TO 221
222   WRITE(NPRNT,110)I,J
221   CONTINUE
C
C                    TERMINAL DOWN
C
      WRITE(NPRNT,112)
112   FORMAT('1',T30,'TERMINAL FAULT PARAMETERS',/,'0 TERM NO',T20,
     * 'MEAN',T40,'VARIANCE',T60,'LENGTH',T80,'VARIANCE OF LENGTH')
      IF(NTM.LE.0) GO TO 240
      DO 231 I=1,NTM
      WRITE(NPRNT,113)I       ,TMEAN(I),TMVAR(I),TLEN(I),TLVAR(I)
113   FORMAT(1X,I4,T20,F10.4,T40,F10.4,T60,F10.4,T80,F10.4)
231   CONTINUE
C
C                    TERMINAL RESPONSE
C
240   CONTINUE
      IF(NTR.LE.0) GO TO 250
      WRITE(NPRNT,114)
114   FORMAT('1',T30,'TERMINAL RESPONSE',/,'0 TERM NO',T20,'RESPONSE',
     * T40,'VARIANCE OF RESPONSE TIME')
      DO 241 I=1,NTR
      WRITE(NPRNT,115)I       ,TRES(I),TRVAR(I)
115   FORMAT(1X,I4,T20,F10.4,T40,F10.4)
241   CONTINUE
C
C                    NOISE
```

```
C
250     CONTINUE
        IF(NNOISE.LE.0) GO TO 260
        WRITE(NPRNT,116)
116     FORMAT('1',T30,'NOISE PARAMETERS',/,'0 BUS CODE',T20,'MEAN',
       1 T40,'VARIANCE',T60,'LENGTH',T80,'VARIANCE OF LENGTH')
        DO 251 I=1,NNOISE
        WRITE(NPRNT,117)NSBUS(I),SMEAN(I),SMVAR(I),SLNG(I),SLVAR(I)
117     FORMAT(1X,I7,T20,F10.4,T40,F10.4,T60,F10.4,T80,F10.4)
251     CONTINUE
C
C                        NUMBER OF FUIS
C
260     CONTINUE
        WRITE(NPRNT,118)NFUIS,FUI
118     FORMAT('0 NFUIS =',I5,' FUI=',F10.5)
C
C                        SETUP ALL EVENTS
C
C                        DEMAND MESSAGES
C
        IF(NDM.LE.0) GO TO 330
        DO 311 I=1,NDM
        ATRIB(1)=TNOW+RNXT(DM(1,I),DM(2,I),I)
        ATRIB(2)=1000.+FLOAT(I)
311     CALL FILEM(1)
C                        TERMINAL DOWN
C
330     CONTINUE
        IF(NTM.LE.0) GO TO 350
        DO 321 I=1,NTM
        ATRIB(1)=TNOW + RNXT(TMEAN(I),TMVAR(I),I)
        ATRIB(2)=600.+FLOAT(I)
321     CALL FILEM(1)
C                        NOISE
350     CONTINUE
        ILBUS=0
        IRBUS=0
        IF(NNOISE.LE.0) GO TO 360
        DO 351 I=1,NNOISE
        ATRIB(1)=TNOW + RNXT(SMEAN(I),SMVAR(I),I)
        ATRIB(2)=400.+ FLOAT(NSBUS(I))
        ATRIB(3)=I
351     CALL FILEM(1)
C                        FUI 1
360     CONTINUE
        IF(NFUI.LE.0) CALL ERROR(13)
        ATRIB(1)=TNOW
        ATRIB(2)=801.
        CALL FILEM(1)
C
C                        INITIALIZATION DONE
C
C       TRACE START TIME
        ATRIB(1)=0.
        ATRIB(2)=0.
        CALL FILEM(1)
C       TRACE END AND TIME
        ATRIB(1)=1.1
        ATRIB(2)=0.
        CALL FILEM(1)
        RETURN
        END
```

```
      [illegible]
      [illegible]
      [illegible]
      [illegible]
     *  [illegible],ARG,TO,[illegible],IOUD),I
      CALL WAT[illegible]
      [illegible]
      CALL TRAN
      [illegible]
      CALL TERMA[illegible]
      [illegible]
      CALL N[illegible]
      [illegible]
      CALL N[illegible]
      [illegible]
      CALL TRAN[illegible]
      [illegible]
      CALL THMIN[illegible]
      [illegible]
      CALL FUI(J)
      [illegible]
      CONTINUE
      [illegible]
      CALL IMARI[illegible]
      RETURN
      END

      SUBROUTINE FUI(IFUI)
C
C     ESTABLISHES THE NEXT FUI ARRIVAL.  THIS REPRESENTS THE FUI CLOCK.
C
      INCLUDE IT
      COMMON /MUXA/FUITT,XFUI,NFUIS,NFIX(25),FIX(25,25),ICURF
      COMMON /MUXT/IBUSY,MSGTYP,MSGNOW
      COMMON /MUXJIT/JIT,FUISTA
C
      FUISTA=TNOW
      IF(IFUI.NE.1) GO TO 50
      FUITT=FUITT+(NFUIS*XFUI)
      X=TNOW+XFUI
C                     CREATE FUIS 2-NFUIS
      IF(NFUIS.LE.1)GO TO 2
      DO 1 I=2,NFUIS
      ATRIB(1)=X
      ATRIB(2)=100.+FLOAT(I)
      CALL FILEM(1)
    1 X=X+XFUI
C                     CREATE NEXT FUI 1
    2 CONTINUE
      ATRIB(1)=TNOW+XFUI*FLOAT(NFUIS)
      ATRIB(2)=101.
      CALL FILEM(1)
C
C                     GENERATE FIRST FIXED MESSAGE IF ONE
C
   50 ICURF=IFUI
      MSGNOW=1
      MSGTYP=1
      JIT=1
      CALL NXTMSG(1)
      RETURN
      END
      SUBROUTINE STAT(ILBUS,IRBUS)
C
C     COLLECTS TIME PERSISTENT STATISTICS ON DATA BUS NOISE.
C
      INCLUDE IT
      X=ILBUS
      CALL TIMST(X,TNOW,1)
      X=IRBUS
      CALL TIMST(X,TNOW,2)
      X=MIN0(ILBUS,IRBUS)
      CALL TIMST(X,TNOW,3)
      RETURN
      END


      SUBROUTINE NXTMSG(IAM)
C
C     SCHEDULES THE NEXT POSSIBLE CALL TO A TERMINAL EVENT.
C
      INCLUDE IT
      COMMON /MUX1/NCM,DM(4,25),TMSENT(25)
      COMMON /MUX2/FUITT,FUI,NFUIS,NFIX(25),FIX(25,25),ICURF
      COMMON /MUX5/NTR,TRES(15),TRVAR(15),NUMV(15),IOK(15)
      COMMON /MUX6/ILBUS,IRBUS
      COMMON /MUX7/IBUSY,MSGTYP,MSGNOW
      COMMON /MUXJIT/JIT,FUISTA
      COMMON /MUXUSE/FUIPRS,FME,DME
      COMMON /STAT/NMSGH,NL,NP,NB,NTO,NMSG
C
C     ARGUMENT IAM HAS FOLLOWING POSSIBLE VALUES AND MEANINGS
C     1 - FUI TIME IS OCURING NOW
C     2 - WATCH DOG TIMEOUT EVENT IS OCCURING NOW
C     3 - A NORMAL RESPONSE IS OCCURING NOW
C
C
C     TEST IF FUI AND BUSY WITH A MESSAGE
C
      IF((IAM.EQ.1).AND.(IBUSY.NE.0))RETURN



C
C     NEED TO SEND A MESSAGE - DETERMINE TYPE
C
      GO TO (10,20),MSGTYP
C
 10   CONTINUE
C     COLLECT JITTER HISTO
      GO TO (1 ,  ),JIT
 2    JIT=2
      X=TNOW-FUISTA
      CALL HISTO(X,1)
      IF((IAM.NE.1).AND.(IBUSY.EQ.1)))DME=TNOW
      IF(ICURF.NE. [illegible] 11
      IF((TNOW-TFUS).LE. FUI/ [illegible] TO 11
C
C     COLLECT STATISTICS
C
      X=FME-FUIPRS
      CALL COLCT(X,1)
      X=DME-FUIPRS
      CALL COLCT(X,2)
 11   FUIPRS=TNOW
 20   CONTINUE
C
C     FIND ME DATA IF POSSIBLE
C
```

```
C
      IF(FIX(MSGNOW,ICURF).LE.0.)GO TO 12
      I=MOD(MSGNOW,5)+1
      X=FLOAT(NTR)+.5
      ATRIB(1)=TNOW+FIX(MSGNOW,ICURF)
      ATRIB(2)=200.0+UNFRM(1.5,X,I)
      ATRIB(3)=MSGNOW+ILBUS*1000.+IRBUS*100000.
      CALL FILEM(1)
C
C     WATCH DOG FOR MESSAGE
C
      I=ATRIB(2)-200.
      X=1.05*(FIX(MSGNOW,ICURF)+TRES(I)+TRVAR(I))
      ATRIB(1)=TNOW+X
      ATRIB(2)=100.
      ATRIB(3)=MSGNOW
      CALL FILEM(1)
      MSGNOW=MSGNOW+1
      IBUSY=1
      NMSG=NMSG+1
      RETURN
C
12    CONTINUE
      MSGTYP=2
      FME=TNOW
      GO TO 1
C
20    CONTINUE
C
C     DEMAND MESSAGE LOOK
C
C
C     TRY TO SEND SOME DEMAND MESSAGES
C
C     X=TIME REMAINING TILL NEXT FUI START
C
      X=(FUI1T+(ICURF*FUI))-TNOW
C
C     SELECT ENTRY FROM QUE
C
      I=MFE(2)
      IF(I.LE.0)GO TO 90
      CALL COPY(I)
      DML=ATRIB(1)
C
      IF(DML.GT.X)GO TO 90
      CALL RMOVE(I,2)
      ATRIB(1)=TNOW+DML
      I=ATRIB(2)
      ATRIB(2)=200.+DM(4,I)
      ATRIB(3)=ILBUS*1000.+IRBUS*100000.+I+200+
      CALL FILEM(1)
C
      TMSENT(ICURF)=TMSENT(ICURF)+DML
C



      J=DM(4,I)
      X=1.05*(DML+TRES(J)+TRVAR(J))
      ATRIB(1)=TNOW+X
      ATRIB(2)=100.
      ATRIB(3)=200.+ I
      CALL FILEM(1)
      IBUSY=1
      NMSG=NMSG+1
      RETURN
90    CONTINUE
      IBUSY=0
      DME=TNOW
      RETURN
      END
      SUBROUTINE CTERM(ITRM       )
C
C     HANDLES MESSAGE ARRIVAL AND SUBSEQUENT PROCESSING AT A TERMINAL
C     AND GENERATES TERMINAL RESPONSE.
C
      INCLUDE IT
      COMMON /MUX5/NTR,TRES(15),TRVAR(15),NUMV(15),IOK(15)
      COMMON /MUX6/ILBUS,IRBUS
      COMMON /STAT/NMSGH,NL,NR,NB,NTO,NMSG
C
C                 MODULE FOR TERMINAL RECEIVING A CALL AND:
C                 A. GENERATING A RESPONSE
C                 B. NO ACTION
C                 DEPENDING ON:
C                 A. TERMINAL IS OPERATIONAL
C                 B. MESSAGE RECEIVE READABLE FROM AT LEAST
C                    ONE BUS
C
C                 STATISTICS COLLECTION IF ANY
C
C                 TEST FOR NOISE HITS
C
      IHTR=ATRIB(3)/100000.+.5
      IHTL=(ATRIB(3)-(IHTR*100000.))/1000.+.5
      MSG=ATRIB(3)-(IHTR*100000.)-(IHTL*1000.)+.5
C
C                 NO RESPONSE IF HIT ON BOTH SIDES OR TERMINAL DOWN
C
      IF((IHTR.GT.0).OR.(IHTL.GT.0))NMSGH=NMSGH+1
      IF((IHTR.GT.0).AND.(IHTL.GT.0))RETURN
      IF(IOK(ITRM).GT.0)RETURN
C
C                 OK - GENERATE RESPONSE
C
      ATRIB(1)=TNOW+RNXT(TRES (ITRM),TRVAR(ITRM),ITRM)
      ATRIB(2)=300.+ ITRM
      ATRIB(3)=MSG+ILBUS*1000.+IRBUS*100000.
      CALL FILEM(1)
      NMSG=NMSG+1
      RETURN
      END
      SUBROUTINE TERMR(ITRM)
C
C     PROCESSES TERMINAL RESPONSE.
C
      INCLUDE IT
      COMMON /MUX2/FUIIT,FUIL,FUIS,NFIX(25),FIX(25,25),ICURF
      COMMON /MUX5/NTR,TRES(15),TRVAR(15),NUMV(15),IOK(15)
      COMMON /MUXRES/IRESE
      COMMON /STAT/NMSGH,NL,NR,NB,NTO,NMSG
C
C                   TERMINAL HAS RESPONDED
C
      IHTR=ATRIB(3)/100000.+.5
      IHTM= ATRIB(3)-(IHTR*100000.))/10.+.5
      IHTL=(ATRIB(3)-(IHTR*100000.))/1000.+.5
      MSG=ATRIB(3)-(IHTR*100000.)-(IHTL*1000.)+.5
C
C                   COLLECT STATISTICS
C
      IF((IHTR.GT.0).OR.(JHTL.GT.0))NMSG=NMSG+1
C
C     TEST IF RESPONSE READABLE
C
      IF((IHTR.GT.0).AND.(IHTL.GT.0))GO TO 90
C
C
C                   KILL WATCH DOG ON THIS MESSAGE
C
      J=0
      X=MSG
1     CONTINUE
      I=NFIND(X,5,1,3,.5)
      IF(I.EQ.0)GO TO 50
      CALL RMOVE(I,1)
      IF(ATRIB(2).EQ.100.)GO TO 10
C     REMOVE TEMPORALLY ANY MESSAGE NO. EVENT NOT WATCH DOG TIMER
C     EVENT.
      CALL FILEM(3)
      J=J+1
      GO TO 1
10    CONTINUE
      IF(J.EQ.0) GO TO 20
      DO 11 I=1,J
      CALL RMOVE(MFE(3),3)
11    CALL FILEM(1)
20    CONTINUE
      CALL NXTMSG(3)
90    CONTINUE
      RETURN
50    IRESE=IRESE+1
      IF(J.NE.0)GO TO 10

      RETURN
      END


      SUBROUTINE WATCH(IDUM)
C
C     WATCH DOG EVENT OCCURENCE IS ESTABLISHED.
C
      IF((IHTR.GT.0).OR.(JHTL.GT.0))NMSG=NMSG+1
C
C     COLLECT STATISTICS ON NO RESPONSE FROM TERMINAL
C
      NTO=NTO+1
      CALL NXTMSG(2)
      RETURN
      END
      SUBROUTINE NOISES(IN)
C
C     PROCESSES NOISE EVENTS AND SCHEDULES NEXT NOISE EVENT AND
C     DURATION.
C
      INCLUDE IT
      COMMON /MUX3/NNOISE,NSBUS(15),SMEAN(15),SMVAR(15),SLNG(15),SLVAR
     1(15)
      COMMON /MUX4/ILBUS,IRBUS
      COMMON /STAT/NMSGH,NL,NR,NB,NTO,NMSG
C
C     NOISE EVENT START
C
      NNO=ATRIB(3)
      N=ATRIB(2)-400
      X=ATRIB(2)
      IF((X.LE.400.).OR.(X.GT.407.))CALL ERROR(8)
      IF(X.EQ.401.)Y=1000.
      IF(X.EQ.402)Y=100000.
      IF(X.EQ.403.)Y=101000.
C
C     MARK EVERY MESSAGE ON BUS (FILE 1) AS HIT
C
1     CONTINUE
      I=NFIND(225,5,1,2,26.)
      IF(I.EQ.0)I=NFIND(325.,5,1,2,26.)
      IF(I.EQ.0) GO TO 10
      CALL RMOVE(I,1)
      CALL FILEM(3)
      GO TO 1
10    CONTINUE
C
C                   MARK AND RETURN TO FILE 1
C
      I=MFE(3)
      IF(I.EQ.0) GO TO 20
      CALL RMOVE(I,3)
      ATRIB(3)=ATRIB(3)+Y
      CALL FILEM(1)
      GO TO 10
20    CONTINUE
      GO TO (21,22,23),IN
21    ILBUS=ILBUS+1
      NL=NL+1
      GO TO 30
22    IRBUS=IRBUS+1
      NR=NR+1
      GO TO 30


  23  ILBUS=ILBUS+1
      IRBUS=IRBUS+1
30    CONTINUE
      IF((ILBUS.GT.0).AND.(IRBUS.GT.0))NB=NB+1
      CALL STAT(ILBUS,IRBUS)
C
C                   NOW FOR NOISE TERMINATION EVENT CREATION
C
      ATRIB(1)=TNOW+RNXT(SLNG(NNO),SLVAR(NNO),NNO)
      ATRIB(2)=500.+IN
      ATRIB(3)=NNO
      CALL FILEM(1)
C
C                   NEXT NOISE EVENT(THIS NUMBER)
C
      ATRIB(1)=TNOW+RNXT(SMEAN(NNO),SMVAR(NNO),NNO)
      ATRIB(2)=400.+IN
      ATRIB(3)=NNO
      CALL FILEM(1)
C
C                   ALL DONE
C
      RETURN
      END
      SUBROUTINE NOISET(IN)
C
C     PROCESSES NOISE TERMINATION EVENT. REMOVES NOISE EVENT(S) FROM
C     BUS(ES).
C
      INCLUDE IT
      COMMON /MUX7/NNOISE,NSBUS(15),SMEAN(15),SMVAR(15),SLNG(15),SLVAR
     1(15)
      COMMON /MUX6/ILBUS,IRBUS,
C
C                    END NOISE EVENT
C
      GO TO(1,2,3),IN
1     ILBUS=ILBUS-1
      GO TO 10
2     IRBUS=IRBUS-1
      GO TO 10
3     IRBUS=IRBUS-1
      ILBUS=ILBUS-1
10    CONTINUE
      CALL STAT(ILBUS,IRBUS)
      RETURN
      END


      FUNCTION RNXT(RMEN,RVAR,ISTRM)
C
C     USING UNFRM DERIVE A NUMBER TO ADD TO TNOW GIVING NEXT
C     EVENT TIME. ISTRM IS ASSUMED ANY POSITIVE INTEGER.
C
      I=MOD(ISTRM,5)+1
      XHI=RVAR
      XLO=-XHI
      RNXT=RMEN+UNFRM(XLO ,XHI,I)
      RETURN
      END

      SUBROUTINE TRMUP(ITN)
C
C     EDSTABLISHES TERMINAL RECOVERY FROM FAILED MODE.
C
      COMMON /MUX5/NTR,TRES(15),TRVAR(15),NUMV(15),IOK(15)
      IOK(ITN)=IOK(ITN)-1
      RETURN
      END

      SUBROUTINE TRMDN(ITN)
C
C     ESTABLISHES TERMINAL FAILURE.
C
      INCLUDE IT
      COMMON /MUX4/NTM,TMEAN(15),TMVAR(15),TLEN(15),TLVAR(15)
      COMMON /MUX5/NTR,TRES(15),TRVAR(15),NUMV(15),IOK(15)
      IOK(ITN)=IOK(ITN)+1
C
C     NEXT DOWN EVENT
C
      ATRIB(1)=TNOW+RNXT(TMEAN(ITN),TMVAR(ITN),ITN)
      ATRIB(2)=600.+ITN
      CALL FILEM(1)
C
C     TERMINAL UP EVENT SCHEDULED NEXT
C
      ATRIB(1)=TNOW+RNXT(TLEN(ITN),TLVAR(ITN),ITN)
      ATRIB(2)=700.+ITN
      CALL FILEM(1)
C
      RETURN
      END
      SUBROUTINE DMARIV(IDM)
C
C     PROCESSES DEMAND MESSAGE ARRIVAL AND SCHEDULES NEXT DEMAND MESSAGE
C     ARRIVAL.
C
      INCLUDE IT
      COMMON /MUX1/NDM,DM(4,25),TMSENT(25)
C
C     DEMAND MESSAGE IDM HAS ARRIVED. PUT ON QUE
C
      ATRIB(1)=DM(3,IDM)
      ATRIB(2)=IDM
      ATRIB(3)=TNOW
C
C     SAVE LENGTH,MESSAGE NUMBER,ARIVAL TIME ON QUE
C
      CALL FILEM(2)
C
C     NEXT D.M. NO. IDM
C
      ATRIB(1)=TNOW+RNXT(DM(1,IDM),DM(2,IDM),IDM)
      ATRIE(2)=1000.+IDM
      CALL FILEM(1)
      RETURN
      END


      SUBROUTINE OTPUT
C
C     OUTPUTS NON-GASP INFORMATION ABOUT THE EFFECT OF BUS NOISE.
C
      INCLUDE IT
      COMMON /MUXREC/IRESE
      COMMON /STAT/NMSGH,NL,NR,NB,NTO,NMSG
      WRITE(NPRNT,100)IRESE
100   FORMAT('1NUMBER OF VALID READABLE RESPONSES LOST TO TIMEOUT =',I9)
      WRITE(NPRNT,101)NTO
101   FORMAT('0NUMBER OF TIMEOUTS =',I9)
      WRITE(NPRNT,102)NL,NR,NP
102   FORMAT('0NUMBER OF HITS ON LEFT BUS =',I9,
     1' RIGHT BUS =',I9,
     2' BOTH BUSES = ',I9)
      WRITE(NPRNT,103)NMSG
103   FORMAT('0TOTAL NUMBER OF MESSAGES SENT =',I9)
      WRITE(NPRNT,104)NMSGH
104   FORMAT('0TOTAL NUMBER OF MESSAGES HIT ON AT LEAST ONE BUS =',I9)
      RETURN
      END
```

FORTRAN IV Simulation Example



```
      INTEGER EVENT
C*************************************************************
C                                                            *
C     INITILIZITION OF ALL VARIABLE AND PARAMETER            *
C                                                            *
C*************************************************************
C                                                            *
C     RANDOM NUMBER GENERATOR SEED                           *
      IX= 13573951373
C                                                            *
C*************************************************************
C                                                            *
C                                                            *
C     SUM OF SERVICE TIME FOR ALL RUNS                       *
      CSOST= 0.0
C                                                            *
C*************************************************************
C                                                            *
C     SUM OF QUEUE LENGTH FOR ALL RUNS                       *
C                                                            *
      CSOQT= 0.0
C                                                            *
C*************************************************************
C                                                            *
C     MEAN INTER ARRIVAL TIME OF MESSAGE                     *
C                                                            *
      EXPA= 5.0
C                                                            *
C*************************************************************
C                                                            *
C     MEAN SERVICE TIME OF MESSAGE                           *
C                                                            *
      EXPS= 4.0
C                                                            *
C*************************************************************
C                                                            *
C     TOTAL SIMULATION TIME                                  *
C                                                            *
      TST= 480.0
C                                                            *
C*************************************************************
C                                                            *
      WRITE(6,1)
      WRITE(6,2)
      DO 100 I=1, 100
C*************************************************************
C                                                            *
C     NUMBER OF MESSAGE ENTRIES                              *
C                                                            *
          TIQ= 0.0
C                                                            *
C*************************************************************
C     FACILITY STATUS                                        *
C                                                            *
          IFS= 0
C                                                            *
C*************************************************************
C     QUEUE LENGTH                                           *
C                                                            *
          IQ= 0
C                                                            *
C*************************************************************
C                                                            *
C     TIME OF LAST QUEUE CHANGE                              *
C                                                            *
          TOLQC= 0.0
C*************************************************************
C                                                            *
          SOST= 0.0
C                                                            *
C*************************************************************
C                                                            *
C     SUM OF QUEUE LENGTH                                    *
```

```
C
          SOQL= 0.0
C                                                                        *
C*************************************************************************
C                                                                        *
C     TIME OF NEXT SERVICE                                               *
C                                                                        *
          TONS= 9999.
C                                                                        *
C*************************************************************************
C                                                                        *
C     TIME OF NEXT DEPARTURE                                             *
C                                                                        *
          TOND= 9999.
C                                                                        *
C*************************************************************************
C                                                                        *
C     TIME OF NEXT ARRIVAL                                               *
C                                                                        *
          TONA= 0.0
C                                                                        *
C*************************************************************************
C                                                                        *
C     SYTEM CLOCK                                                        *
          CLOCK= 0.0
C                                                                        *
C*************************************************************************
          CALL RAND(N,R)
          TONA= -EXPA*ALOG(R)
200     CONTINUE
          IF (TONA .GT. TST) GOTO 40
          ASSIGN 10 TO EVENT
          IF (TONS .LT. TONA) ASSIGN 20 TO EVENT
          IF (TOND .LT. TONS .AND. TOND .LT. TONA) ASSIGN 30 TO EVENT
          GOTO EVENT,(10,20,30)
10      CONTINUE
          CALL ARIVAL(TONA,TONS,CLOCK,TOLQC,IG,TIG,IFS,SOQL,EXPA,N)
          GOTO 200
20      CONTINUE
          CALL SERV(TONS,TOND,CLOCK,TOLQC,IG,IFS,SOQL,SOST,EXPS,N)
          GOTO 200
30      CONTINUE
          CALL DEPART(TOND,TONS,CLOCK,IG,IFS)
          GOTO 200
40      CONTINUE
C*************************************************************************
C                                                                        *
C     CALCULATION OF AVERAGE QUEUE LEENGTH(AVQL),AVERAGE                 *
C     SERVICE TIME(AVST)                                                 *
C                                                                        *
C*************************************************************************
          AVQL= SOQL/CLOCK
          AVST= SOST/CLOCK
          SSOQT= SSOQT+AVQL
          SSOST= SSOST+AVST
        WRITE(6,3) AVQL,AVST
100     CONTINUE
C*************************************************************************
C                                                                        *
C     CALCULATION OF AVERAGE QUFUE FOR ALL RUN,AVERARAGE                 *
C     UTILIZATION FOR ALL RUN AND AVERAGE TIME OF  MESSAGE ENTRY FOR     *
C     ALL RUN                                                            *
C                                                                        *
C*************************************************************************
         AVQL= SSOQT/100.
         AVST= SSOST/100.
        WRITE(6,4) AVQL,AVST
1       FORMAT('1',/////,30X,'S I M U L A T I O N   S T A T I S T I C S')
2       FORMAT('0',15X,'AVE QUEUE LENGHT',20X,'AVE UTILIZATION')
3       FORMAT('0'17X,F9.4,25X,F8.4)
4       FORMAT('0',20X,'AVE QUEUE LENGHT FOR ALL RUNS = ',F9.4,
       */'0',20X,'AVE UTILIZATION FOR ALL RUNS = ',F8.4)
        STOP
        END
```

```
SUBROUTINE ARIVAL(TONA,TONS,CLOCK,TOLQC,IQ,TIQ,IFS,SOQL,EXPA,N)
   CLOCK= TONA
   SOQL= SOQL+(CLOCK-TOLQC)*IQ
   TOLQC= CLOCK
   TIQ= TIQ+ 1.
   IQ= IQ+1
   CALL RAND(N,R)
   TONA= CLOCK+(-EXPA*ALOG(R))
   IF (IFS .NE. 0 .OR. IQ .GT. 1) RETURN
   TONS= CLOCK
RETURN
END
```

```
SUBROUTINE SERV(TONS,TOND,CLOCK,TOLQC,IQ,IFS,SOQL,SOST,EXPS,N)
   CLOCK= TONS
   CALL RAND(N,R)
   SERVT= -EXPS*ALOG(R)
   TOND= CLOCK+SERVT
   SOST= SOST+SERVT
   SOQL= SOQL+(CLOCK-TOLQC)*IQ
   TOLQC= CLOCK
   IQ= IQ-1
   IFS= 1
   TONS= 9999.
RETURN
END
```

```
SUBROUTINE DEPART(TOND,TONS,CLOCK,IQ,IFS)
   CLOCK= TOND
   IF (IQ .GT. 0) TONS= CLOCK
   TOND= 99999.
   IFS= 0
RETURN
END
```

```
SUBROUTINE RAND(N,R)
   K= 315227
   N= N*K
   RN= N
   R= RN/34359738337.
   R= ABS(R)
RETURN
END
```

SIMSCRIPT II Simulation Example

```
PREAMBLE''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

''SIMULATION PROGRAM FOR SINGLE QUEUE SINGLE SERVER TIME OF MESSAGE
''ARRIVAL EXPONENTIALY DISTRIBUTED WITH MEAN 5 AND  SERVICE TIME IS
''EXPONENTIALY DISTRIBUTED WITH MEAN 4
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
''
''
    EVENT NOTICES INCLUDE ARRIVAL, DEPARTURE AND STOP.SIMULATION
    TEMPORARY ENTITIES
       EVERY MESSAGE  MAY BELONG TO THE QUEUE
''
    DEFINE SERVICE.TIME, QUEUE.CHANGE, SUM.QUEUE AS VARIABLE
    DEFINE STATUS AS AN INTEGER VARIABLE
    DEFINE IDLE TO MEAN 0
    DEFINE BUSY TO MEAN 1
    DEFINE TOTAL.ENTRY AS AN INTEGER VARIABLE
''
    THE SYSTEM OWNS THE QUEUE
    TALLY MAX.QUEUE AS THE MAXIMUM OF N.QUEUE
''
    ACCUMULATE UTILIZATION AS THE AVERAGE OF STATUS
END
```

```
MAIN
    SCHEDULE A ARRIVAL IN EXPONENTIAL.F(5.0,1) MINUTES
    SCHEDULE A STOP.SIMULATION IN 8 HOURS
    START SIMULATION
END

EVENT ARRIVAL
    SCHEDULE A ARRIVAL IN EXPONENTIAL.F(5.0,1) MINUTES
    CREATE A MESSAGE
''
    IF STATUS = BUSY OR N.QUEUE > 1,
        LET SUM.QUEUE= SUM.QUEUE+(TIME.V*1440.-QUEUE.CHANGE)*N.QUEUE
''
        FILE MESSAGE  IN QUEUE
    LET TOTAL.ENTRY=TOTAL.ENTRY+1
''
        LET QUEUE.CHANCE= TIME.V*1440.
        RETURN
    ELSE
        LET SERVICE.TIME= EXPONENTIAL.F(4.0,3)
''
        SCHEDULE A DEPARTURE IN SERVICE.TIME MINUTES
        LET STATUS= BUSY
        RETURN
END
```

```
EVENT DEPARTURE
    IF QUEUE IS EMPTY,
        LET STATUS= IDLE
        RETURN
    ELSE
        LET SUM.QUEUE= SUM.QUEUE+(TIME.V*1440.-QUEUE.CHANGE)*N.QUEUE
''
        REMOVE FIRST MESSAGE  FROM QUEUE
        LET QUEUE.CHANGE= TIME.V*1440.
''
        DESTROY MESSAGE
''
        LET SERVICE.TIME= EXPONENTIAL.F(4.0,3)
        SCHEDULE A DEPARTURE IN SERVICE.TIME MINUTES
        RETURN
END
```

```
EVENT STOP.SIMULATION
    LET AVE.QUEUE.LENGTH= SUM.QUEUE/(TIME.V*1440.)
''
    START NEW PAGE
    SKIP 5 LINES
    PRINT 7 LINE WITH   AVE.QUEUE.LENGTH,UTILIZATION,MAX.QUEUE
    AND TOTAL.ENTRY THUS

 S I M U L A T I O N   S T A T I S T I C S

    AVERAGE QUEUE LENGTH    *.**
    UTILIZATION             *.** MINUTES
   MAX.QUEUE                ***
    TOTAL.ENTRY             *** MESSAGE ENTRY
    STOP
END
```

GASP IV Simulation Example

```
SC 12:44:43 (->0)        RI
   1T      PROC
         DIMENSION NSET(1)
         COMMON QSET(5000)
         EQUIVALENCE (NSET(1),QSET(1))
         COMMON/GCOM1/ATRIB(25),JEVNT,MFA,MFE(100),MLE(100),MSTOP,NCRDR,
        &NNAPO,NNAPT,NNATR,NNFIL,NNQ(100),NNTRY,NPRNT,PPARM(50,4),TNOW,
        &TTBEG,TTCLR,TTFIN,TTRIB(25),TTSET
         COMMON /GCOM2/ DD(100),DDL(100),DTFUL,DTNOW,ISEES,LFLAG(50),NFLAG,
        &NNEQD,NNEQS,NNEQT,SS(100),SSL(100),TTNEX
         COMMON/GCOM3/ AAERR,DTMAX,DTMIN,DTSAV,IITES,LLERR,LLSAV,LLSEV,PRE
        &PR,TTLAS,TTSAV
         COMMON/GCOM4/ DTPLT(10),HHLOW(25),HHWID(25),IICRD,IITAP(10),JJCEL
        &(500),LLABC(25,2),LLABH(25,2),LLABP(11,2),LLABT(25,2),LLPHI(10),LL
        &PLO(10),LLPLT,LLSUP(15),LLSYM(10),MMPTS,NNCEL(25),NNCLT,NNHIS,NNPL
        &T,NNPTS(10),NNSTA,NNVAR(10),PPHI(10),PPLO(10)
         COMMON/GCOM5/ IIEVT,IISED(6),JJBEG,JJCLR,MMNIT,MMON,NNAME(3),NNCF
        &I,NNDAY,NNPT,NNSET,NNPRJ,NNPRM,NNPNS,NNRUN,NNSTR,NNYR,SSEED(6)
         COMMON/GCOM6/EENQ(100),IINN(100),KKRNK(100),MMAXQ(100),QQTIM(100)
        &,SSOEV(25,5),SSTPV(25,6),VVNQ(100)
    END


C***************************************************************
C                                                              *
C      MAIN PROGRAM                                            *
C                                                              *
C***************************************************************
       INCLUDE IT
C***************************************************************
C                                                              *
C      INITIALIZE CARD READER VALUE,NCRDR AND PRINTER VALUE,NPRNT .   *
C                                                              *
C***************************************************************
       NCRDR=5
       NPRNT=6
       CALL GASP
       STOP
       END


       SUBROUTINE EVNTS(IX)
       INCLUDE IT
C***************************************************************
C                                                              *
C      IF IX IS 20 EVENT ARRIVAL WILL OCCUR.                   *
C      IF IX IS 30 EVENT BEGIN SERVICE WILL OCCUR.             *
C      IF IX IS 40 EVENT EVENT FINISH SERVICE WILL OCCUR.      *
C                                                              *
C***************************************************************
       GO TO (20,30,40),IX
   20  CALL ARR
       RETURN
   30  CALL BEGS
       RETURN
   40  CALL FINS
       RETURN
       END
```

```
      SUBROUTINE ARR
      INCLUDE IT
C***************************************************************
C                                                              *
C     FILE 2 = QUEUE                                           *
C     ATTRIBUTE 1 = TIME OF ARRIVAL                            *
C                                                              *
C


C***************************************************************
      ATRIB(1)=TNOW
C***************************************************************
C                                                              *
C     SCHEDULE NEXT EVENT                                      *
C                                                              *
C***************************************************************
      CALL FILEM(2)
C***************************************************************
C                                                              *
C     FILE 3 SERVICE FACILITY                                  *
C     SCHEDULE NEXT ARRIVAL MESSAGE                            *
C                                                              *
C***************************************************************
      X=DRAND(1)
      ATRIB(1)= TNOW + ((-5.)*ALOG(X))
C***************************************************************
C                                                              *
C     CODE FOR SERVICE ARRIVAL=1                               *
C                                                              *
C***************************************************************
      ATRIB(2)=1
      CALL FILEM(1)
C***************************************************************
C                                                              *
C TEST WETHER THERE IS ANY MESSAGE IN THE QUEUE AND SERVICE  FACILITY *
C     IS FREE                                                  *
C     NNQ(2) NUMBER OF MESSAGE INTRY IN FILE 2                 *
C                                                              *
C***************************************************************
      IF(NNQ(2).EQ.1.AND. NNQ(3) .EQ. 0) GO TO 10
      RETURN
C***************************************************************
C                                                              *
C     SCHEDULE NEXT EVENT                                      *
C                                                              *
C***************************************************************
   10 ATRIB(1)= TNOW
C***************************************************************
C                                                              *
C     CODE FOR  BEGINNING OF SERVICE=2                         *
C                                                              *
C***************************************************************
      ATRIB(2)=2
      CALL FILEM(1)
      RETURN
      END
      SUBROUTINE [illegible]
      INCLUDE [illegible]
C**************************************************************
C                                                             *
C     REMOVE MESSAGE FORM QUEUE                               *
C     MFE(1) IS RELATIVE ADDRESS OF FIRST MESSAGE ENTRY IN FILE 1
C                                                             *
C**************************************************************
      CALL RMOVE(MFE(1),1)
C**************************************************************
C                                                             *
C     PUT MESSAGE INTO SERVICE FACILITY                       *
C     SET ATRIBUTE 2S TIME OF BEGIN SERVICE                   *
C                                                             *
C**************************************************************
      ATRIB(2)=TNOW
      CALL FILEM(7)
C**************************************************************
C                                                             *
C     SCHEDULE FINSHEING SERVICE                              *
C                                                             *
C**************************************************************
      X=DRAND(1)
      ATRIB(1)=TNOW+(-4.)*ALOG(X))
C**************************************************************
C                                                             *
C     CODE FOR FINISH SERVICE=3                               *
C                                                             *
C**************************************************************
      ATRIB(2)=3
C**************************************************************
C                                                             *
C     SCHEDULE NEXT EVENT                                     *
C                                                             *
C**************************************************************
      CALL FILEM(1)
      RETURN
      END
      SUBROUTINE FINS
      INCLUDE [illegible]
C**************************************************************
C                                                             *
C     REMOVE MESSAGE FROM SERVICE FACILITY                    *
C     MLE(7) IS RELATIVE ADDRESS OF LAST MESSAGE ENTRY IN FILE 7
C     FILE 7 IS SERVICE FACILITY                              *
C                                                             *
C**************************************************************
      CALL RMOVE(MLE(7),7)
      X=TNOW-ATRIB(2)
C**************************************************************
C                                                             *
C     SCHEDULE BEGIN SERVICE IF THERE IS NO MESSAGE IN THE QUEUE
C     NNQ(1) IS CURRENT NO. OF MESSAGE ENTRY IN FILE 1 (QUEUE)
C                                                             *
C**************************************************************
      IF(NNQ(1).EQ.0) RETURN
      ATRIB(1)=TNOW
      ATRIB(2)=2
C**************************************************************
C                                                             *
C     SCHEDULE NEXT EVENT                                     *
C                                                             *
C**************************************************************
      CALL FILEM(1)
      RETURN
      END
```

GPSS Simulation Example

```
LOC  NAME          X      Y      Z      SEL    NBA    NBB    MEAN   MOD    REMARKS    F


***********************************************************************************
**SIMULATION PROGRAM FOR SINGLE QUEUE SINGLE SFRVER TIME OF MESSAGE
**ARRIVAL EXPONENTIALY DISTRIBUTED WITH MEAN 5 AND  SERVICE TIME IS
**EXPONENTIALY DISTRIBUTED WITH MEAN 4


     JOB
     ASSEMBLER
LOC  NAME          X      Y      Z      SEL    NBA    NBB    MEAN   MOD    REMARKS

@    ASSEMBLER
.    FUNCTION      RN1    C24
0    0       .1    .104   .2     .222   .3     .355   .4     .509   .5     .69
.6   .915    .7    1.2    .75    1.38   .8     1.6    .84    1.83   .88    2.12
.9   2.3     .922  2.52   .94    2.81   .95    2.99   .96    3.2    .97    3.5
.98  3.9     .99   4.6    .995   5.3    .998   6.7    .999   7      .9997  8
*
 10  GENERATE                                  11            5      FN2
*
 11  QUEUE         1                           12
*
 12  SEIZE         1                           13
 13  ADVANCE                                   14            4      FN2
*
 14  RELEASE       1
 15  TERMINATE     R                           15
*
     START         0, ,480

     END
```

Plotting Routines

```
      DIMENSION QUE(102), USED(102), FREE(102),TIME(102)
      N=100
      XMAX=-1.E+35
      DO 10 I=1,100
      I1=I-1
      TIME(I)=1.0+.1*I1
      XMAX=AMAX1(XMAX,TIME(I))
10    CONTINUE
      DO 30 I= 1,100
      READ(5,25)  QUE(I),USED(I),FREE(I)
25    FORMAT(1X,3E8.4)
   30 CONTINUE
      CALL RFRD(TIME,FREE,N,XMAX)
      DO 50 I=1,100
      WRITE(6,40) I,TIME(I),QUE(I), USED(I), FREE(I)
40    FORMAT(2X,I3,4(5X,E8.4))
50    CONTINUE
      STOP
      END


       SUBROUTINE RFRD (XX,YY,N,XMAX)
       DIMENSION   XX(102),YY(102)
       XCRG=1.5
       YCRG=1.75
       XSZ=11.00
       YSZ=8.5
       CALL PLOTS(XSZ,YSZ,0,1)
       CALL LINEWT(-1)
       CALL PLOT(1.,1.,-3)
       CALL PLOT(XSZ,0.0,2)
       CALL PLOT(0.0,YSZ,2)
       CALL PLOT(0.0,0.0,2)
       CALL PLOT(XCRG,YORG,-3)
       XCOR=8.
       YCOR=5.
       K=1
       CALL SCALE (XX(K),XCOR,N,1)
       CALL SCALE (YY(K),YCOR,N,1)
       N1=N+1
       N2=N+2
       XX(N2)=XMAX/XCOR
       CALL AXIS(0.0,0.0,5HFREE ,+5,YCOR,90.,YY(N1),YY(N2))
       CALL AXIS(0.0,0.0,4HTIME,-4,XCOR,0.0,XX(N1),XX(N2))
       CALL LINEWT(0)
       CALL LINE(XX(K),YY(K),N,1,0,12,.075)
       CALL PLOT(0.0,0.0,999)
       RETURN
       END
```

REFERENCES

1. Calhoun, M. D., "A Study of two Avionics Multiplex Simulation Models", USAF, August 1979.

2. Multiplex System Simulator (MUXSIM) User's Manual, Contract AFAL F-33615-73-C-1172, Harris Corporation, Melbourne, Florida, June 1976.

3. Shannon, R. E., System Simulation, the Art and Science, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1975.

4. Kiviat, P. J., R. Villanueva, and H. M. Morkkowitz, The SIMSCRIPT II Programming Language, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1968.

5. Brown, L., "Simulation and SIMSCRIPT II.5", Class Notes, CS8553, MSU, April 1980.

6. Pritsker, A. A. B., The GASP IV Simulation Language, J. Willey and Son, New York, 1974.

7. Computer Science Department, GPSS II Programmer's Reference, MSU, November 1978.

8. Sigplan Notices, "Preliminary ADA Reference Manual", Volume 14, Number 6, June 1979, Part A.

9. Kosy, D. W., The ECSS II Language for Simulating Computer Systems, Rand Corporation, 1975.

10. System Modification Design Data Manual, Volume II, Contract AFAL F-33615-73-C-1172, Harris Corporation, Melbourne, Florida, June 1976.

11. Multiplex Simulation Design Study, Contract AFAL F-33615-73-C-1172, Harris Corporation, Melbourne, Florida, June 1976.

END

DATE FILMED

5 81

DTIC